10ª EDIÇÃO

SISTEMAS DIGITAIS

princípios e aplicações

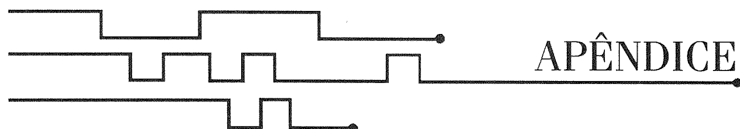


RONALD J. TOCCI | NEAL S. WIDMER | GREGORY L. MOSS





Site com material de apoio para professores e alunos



Folhas de Características do Fabricante de CI*

Estas folhas de características estão apresentadas por cortesia da Texas Instruments Incorporated.

74LS00	**74LS14**	**7432**	**74ALS86**	**74LS193**
74HCT02	**74ALS20**	**74S32**	**74AS86**	**74ALS193**
74LS04	**74AS20**	**74LS37**	**74LS112**	**74293**
7406	**74LS20**	**7486**	**74S112**	**4001B**
7407	**74HC20**	**74LS86**	**74121**	

*As folhas de características reproduzidas a seguir são fac-símiles do manual do fabricante. Na sua vida profissional, você vai encontrar os manuais em inglês. Por isso, foram mantidas aqui na sua forma original. (*N.T.*)

SN54LS00, SN74LS00
QUADRUPLE 2-INPUT POSITIVE-NAND GATES

recommended operating conditions

	SN54LS00			SN74LS00			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC} Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH} High-level input voltage	2			2			V
V_{IL} Low-level input voltage			0.7			0.8	V
I_{OH} High-level output current			− 0.4			− 0.4	mA
I_{OL} Low-level output current			4			8	mA
T_A Operating free-air temperature	− 55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS †	SN54LS00			SN74LS00			UNIT
		MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN, I_I = − 18 mA			− 1.5			− 1.5	V
V_{OH}	V_{CC} = MIN, V_{IL} = MAX, I_{OH} = − 0.4 mA	2.5	3.4		2.7	3.4		V
V_{OL}	V_{CC} = MIN, V_{IH} = 2 V, I_{OL} = 4 mA		0.25	0.4		0.25	0.4	V
	V_{CC} = MIN, V_{IH} = 2 V, I_{OL} = 8 mA					0.35	0.5	
I_I	V_{CC} = MAX, V_I = 7 V			0.1			0.1	mA
I_{IH}	V_{CC} = MAX, V_I = 2.7 V			20			20	μA
I_{IL}	V_{CC} = MAX, V_I = 0.4 V			− 0.4			− 0.4	mA
I_{OS}§	V_{CC} = MAX	− 20		− 100	− 20		− 100	mA
I_{CCH}	V_{CC} = MAX, V_I = 0 V		0.8	1.6		0.8	1.6	mA
I_{CCL}	V_{CC} = MAX, V_I = 4.5 V		2.4	4.4		2.4	4.4	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C
§ Not more than one output should be shorted at a time, and the duration of the short-circuit should not exceed one second.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 2)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Y	R_L = 2 kΩ, C_L = 15 pF		9	15	ns
t_{PHL}					10	15	ns

NOTE 2: Load circuits and voltage waveforms are shown in Section 1.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

SN54HCT02, SN74HCT02
QUADRUPLE 2-INPUT POSITIVE-NOR GATES

SCLS065A – NOVEMBER 1988 – REVISED JANUARY 1996

recommended operating conditions

			SN54HCT02			SN74HCT02			UNIT
			MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage		4.5	5	5.5	4.5	5	5.5	V
V_{IH}	High-level input voltage	V_{CC} = 4.5 V to 5.5 V	2			2			V
V_{IL}	Low-level input voltage	V_{CC} = 4.5 V to 5.5 V	0		0.8	0		0.8	V
V_I	Input voltage		0		V_{CC}	0		V_{CC}	V
V_O	Output voltage		0		V_{CC}	0		V_{CC}	V
t_t	Input transition (rise and fall) time		0		500	0		500	ns
T_A	Operating free-air temperature		–55		125	–40		85	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS		V_{CC}	T_A = 25°C			SN54HCT02		SN74HCT02		UNIT
				MIN	TYP	MAX	MIN	MAX	MIN	MAX	
V_{OH}	$V_I = V_{IH}$ or V_{IL}	I_{OH} = –20 μA	4.5 V	4.4	4.499		4.4		4.4		V
		I_{OH} = –4 mA		3.98	4.3		3.7		3.84		
V_{OL}	$V_I = V_{IH}$ or V_{IL}	I_{OL} = 20 μA	4.5 V		0.001	0.1		0.1		0.1	V
		I_{OL} = 4 mA			0.17	0.26		0.4		0.33	
I_I	$V_I = V_{CC}$ or 0		5.5 V		±0.1	±100		±1000		±1000	nA
I_{CC}	$V_I = V_{CC}$ or 0, I_O = 0		5.5 V			2		40		20	μA
ΔI_{CC}‡	One input at 0.5 V or 2.4 V, Other inputs at 0 or V_{CC}		5.5 V		1.4	2.4		3		2.9	mA
C_i			4.5 V to 5.5 V		3	10		10		10	pF

‡ This is the increase in supply current for each input that is at one of the specified TTL voltage levels rather than 0 V or V_{CC}.

switching characteristics over recommended operating free-air temperature range, C_L = 50 pF (unless otherwise noted) (see Figure 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC}	T_A = 25°C			SN54HCT02		SN74HCT02		UNIT
				MIN	TYP	MAX	MIN	MAX	MIN	MAX	
t_{pd}	A or B	Y	4.5 V		11	20		30		25	ns
			5.5 V		10	18		27		22	
t_t		Y	4.5 V		9	15		22		19	ns
			5.5 V		8	14		20		17	

operating characteristics, T_A = 25°C

	PARAMETER	TEST CONDITIONS	TYP	UNIT
C_{pd}	Power dissipation capacitance per gate	No load	20	pF



SN54LS04, SN74LS04
HEX INVERTERS

recommended operating conditions

		SN54LS04			SN74LS04			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.7			0.8	V
I_{OH}	High-level output current			– 0.4			– 0.4	mA
I_{OL}	Low-level output current			4			8	mA
T_A	Operating free-air temperature	– 55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS †	SN54LS04			SN74LS04			UNIT
		MIN	TYP ‡	MAX	MIN	TYP ‡	MAX	
V_{IK}	V_{CC} = MIN, I_I = – 18 mA			– 1.5			– 1.5	V
V_{OH}	V_{CC} = MIN, V_{IL} = MAX, I_{OH} = – 0.4 mA	2.5	3.4		2.7	3.4		V
V_{OL}	V_{CC} = MIN, V_{IH} = 2 V, I_{OL} = 4 mA		0.25	0.4			0.4	V
	V_{CC} = MIN, V_{IH} = 2 V, I_{OL} = 8 mA					0.25	0.5	
I_I	V_{CC} = MAX, V_I = 7 V			0.1			0.1	mA
I_{IH}	V_{CC} = MAX, V_I = 2.7 V			20			20	μA
I_{IL}	V_{CC} = MAX, V_I = 0.4 V			– 0.4			– 0.4	mA
I_{OS} §	V_{CC} = MAX	– 20		– 100	– 20		– 100	mA
I_{CCH}	V_{CC} = MAX, V_I = 0 V		1.2	2.4		1.2	2.4	mA
I_{CCL}	V_{CC} = MAX, V_I = 4.5 V		3.6	6.6		3.6	6.6	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time, and the duration of the short-circuit should not exceed one second.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 2)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	A	Y	R_L = 2 kΩ,	C_L = 15 pF		9	15	ns
t_{PHL}						10	15	ns

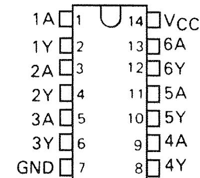
NOTE 2: Load circuits and voltage waveforms are shown in Section 1.

SN5406, SN5416, SN7406, SN7416 HEX INVERTER BUFFERS/DRIVERS WITH OPEN-COLLECTOR HIGH-VOLTAGE OUTPUTS

DECEMBER 1983 – REVISED MARCH 1988

- Converts TTL Voltage Levels to MOS Levels
- High Sink-Current Capability
- Input Clamping Diodes Simplify System Design
- Open-Collector Driver for Indicator Lamps and Relays
- Inputs Fully Compatible with Most TTL Circuits

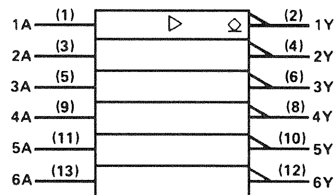
SN5406, SN5416 . . . J OR W PACKAGE
SN7406, SN7416 . . . N PACKAGE
(TOP VIEW)



description

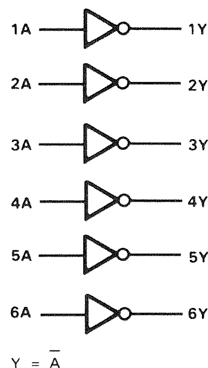
These monolithic TTL hex inverter buffers/drivers feature high-voltage open-collector outputs for interfacing with high-level circuits (such as MOS), or for driving high-current loads (such as lamps or relays), and are also characterized for use as inverter buffers for driving TTL inputs. The SN5406 and SN7406 have minimum breakdown voltages of 30 volts and the SN5416 and SN7416 have minimum breakdown voltages of 15 volts. The maximum sink current is 30 milliamperes for the SN5406 and SN5416, and 40 milliamperes for the SN7406 and SN7416.

logic symbol†



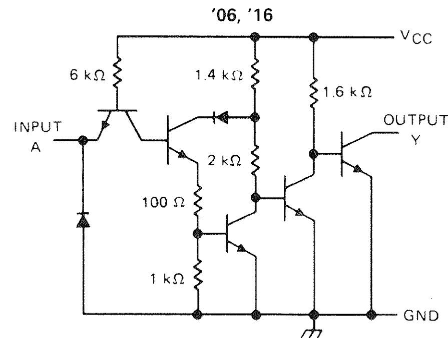
† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

logic diagram (positive logic)



$Y = \overline{A}$

schematic



Resistor values shown are nominal.

SN5406, SN5416, SN7406, SN7416 HEX INVERTER BUFFERS/DRIVERS WITH OPEN-COLLECTOR HIGH-VOLTAGE OUTPUTS

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1) . . . . 7 V
Input voltage (see Note 1) . . . . 5.5 V
Output voltage (see Notes 1 and 2): SN5406, SN7406 Circuits . . . . 30 V
SN5416, SN7416 Circuits . . . . 15 V
Operating free-air temperature range: SN5406, SN5416 Circuits . . . . – 55°C to 125°C
SN7406, SN7416 Circuits . . . . 0°C to 70°C
Storage temperature range . . . . – 65°C to 150°C

NOTES: 1. Voltage values are with respect to network ground terminal.
2. This is the maximum voltage which should be applied to any output when it is in the off state.

recommended operating conditions

		SN5406 SN5416 MIN	NOM	MAX	SN7406 SN7416 MIN	NOM	MAX	UNIT
V_{CC} Supply voltage		4.5	5	5.5	4.75	5	5.25	V
V_{IH} High-level input voltage		2			2			V
V_{IL} Low-level input voltage				0.8			0.8	V
V_{OH} High-level output voltage	'06			30			30	V
	'16			15			15	
I_{OL} Low-level output current				30			40	mA
T_A Operating free-air temperature		–55		125	0		70	C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†		SN5406 SN5416 MIN	TYP‡	MAX	SN7406 SN7416 MIN	TYP‡	MAX	UNIT
V_{IK}	V_{CC} = MIN, I_I = – 12 mA				– 1.5			– 1.5	V
I_{OH}	V_{CC} = MIN, V_{IL} = 0.8 V, V_{OH} = §				0.25			0.25	mA
V_{OL}	V_{CC} = MIN, V_{IH} = 2 V	I_{OL} = 16 mA			0.4			0.4	V
		I_{OL} = ¶			0.7			0.7	
I_I	V_{CC} = MAX, V_I = 5.5 V				1			1	mA
I_{IH}	V_{CC} = MAX, V_{IH} = 2.4 V				40			40	μA
I_{IL}	V_{CC} = MAX, V_{IL} = 0.4 V				– 1.6			– 1.6	mA
I_{CCH}	V_{CC} = MAX			30	48		30	48	mA
I_{CCL}	V_{CC} = MAX			32	51		32	51	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ V_{OH} = 30 V for '06 and 15 V for '16.
¶ I_{OL} = 30 mA for SN54' and 40 mA for SN74'.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 3)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	A	Y	R_L = 110 Ω C_L = 15 pF		10	15	ns
t_{PHL}					15	23	ns

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

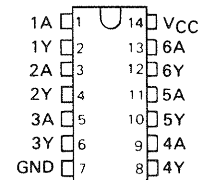
TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

SN5407, SN5417, SN7407, SN7417
HEX BUFFERS/DRIVERS WITH
OPEN-COLLECTOR HIGH-VOLTAGE OUTPUTS

DECEMBER 1983 – REVISED MARCH 1988

- Converts TTL Voltage Levels to MOS Levels
- High Sink-Current Capability
- Input Clamping Diodes Simplify System Design
- Open-Collector Driver for Indicator Lamps and Relays
- Inputs Fully Compatible with Most TTL Circuits

SN5407, SN5417 . . . J OR W PACKAGE
SN7407, SN7417 . . . N PACKAGE
(TOP VIEW)

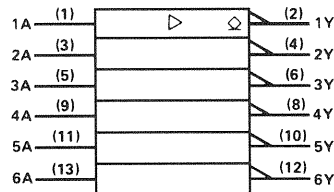


description

These monolithic TTL hex buffers/drivers feature high-voltage open-collector outputs for interfacing with high-level circuits (such as MOS), or for driving high-current loads (such as lamps or relays), and are also characterized for use as buffers for driving TTL inputs. The SN5407 and SN7407 have minimum breakdown voltages of 30 volts and the SN5417 and SN7417 have minimum breakdown voltages of 15 volts. The maximum sink current is 30 milliamperes for the SN5407 and SN5417, and 40 milliamperes for the SN7407 and SN7417.

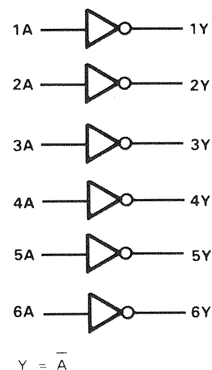
These circuits are completely compatible with most TTL families. Inputs are diode-clamped to minimize transmission-line effects which simplifies design. Typical power dissipation is 145 milliwatts and average propagation delay time is 14 nanoseconds. The SN5407 and SN5417 are characterized for operation over the full military temperature range of −55 C to 125 C; the SN7407 and SN7417 are characterized for operation from 0 C to 70 C.

logic symbol†



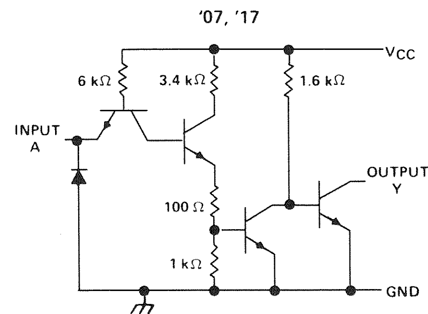
† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

logic diagram (positive logic)



$Y = \overline{A}$

schematic



Resistor values shown are nominal.

TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

SN5407, SN5417, SN7407, SN7417 HEX BUFFERS/DRIVERS WITH OPEN-COLLECTOR HIGH-VOLTAGE OUTPUTS

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage V_{CC} (see Note 1) .... 7 V
Input voltage (see Note 1) .... 5.5 V
Output voltage (see Notes 1 and 2): SN5407, SN7407 Circuits .... 30 V
SN5417, SN7417 Circuits .... 15 V
Operating free-air temperature range: SN5405, SN5417 Circuits .... −55°C to 125°C
SN7407, SN7417 Circuits .... 0°C to 70°C
Storage temperature range .... −65°C to 150°C

NOTES: 1. Voltage values are with respect to network ground terminal.
2. This is the maximum voltage which should be applied to any output when it is in the off state.

recommended operating conditions

		SN5407 SN5417			SN7407 SN7417			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC} Supply voltage		4.5	5	5.5	4.75	5	5.25	V
V_{IH} High-level input voltage		2			2			V
V_{IL} Low-level input voltage				0.8			0.8	V
V_{OH} High-level output voltage	'07			30			30	V
	'17			15			15	
I_{OL} Low-level output current				30			40	mA
T_A Operating free-air temperature		−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†		SN5407 SN5417			SN7407 SN7417			UNIT
			MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN, I_I = −12 mA				−1.5			−1.5	V
I_{OH}	V_{CC} = MIN, V_{IL} = 0.8 V, V_{OH} = §				0.25			0.25	mA
V_{OL}	V_{CC} = MIN, V_{IH} = 2 V	I_{OL} = 16 mA			0.4			0.4	V
		I_{OL} = ¶			0.7			0.7	
I_I	V_{CC} = MAX, V_I = 5.5 V				1			1	mA
I_{IH}	V_{CC} = MAX, V_{IH} = 2.4 V				40			40	μA
I_{IL}	V_{CC} = MAX, V_{IL} = 0.4 V				−1.6			−1.6	mA
I_{CCH}	V_{CC} = MAX			29	41		29	41	mA
I_{CCL}	V_{CC} = MAX			21	30		21	30	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ V_{OH} = 30 V for '07 and 15 V for '17.
¶ I_{OL} = 30 mA for SN54' and 40 mA for SN74'.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 3)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	A	Y	R_L = 110 Ω, C_L = 15 pF		6	15	ns
t_{PHL}					20	26	
t_{PLH}	A	Y	R_L = 150 Ω, C_L = 50 pF			15	ns
t_{PHL}						26	

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

SN54LS14, SN74LS14
HEX SCHMITT-TRIGGER INVERTERS

recommended operating conditions

		SN54LS14			SN74LS14			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
I_{OH}	High-level output current			−0.4			−0.4	mA
I_{OL}	Low-level output current			4			8	mA
T_A	Operating free-air temperature	−55		125	0		70	C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†			SN54LS14			SN74LS14			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{T+}	$V_{CC} = 5$ V			1.4	1.6	1.9	1.4	1.6	1.9	V
V_{T-}	$V_{CC} = 5$ V			0.5	0.8	1	0.5	0.8	1	V
Hysteresis ($V_{T+} - V_{T-}$)	$V_{CC} = 5$ V			0.4	0.8		0.4	0.8		V
V_{IK}	V_{CC} = MIN,	$I_I = -18$ mA				−1.5			−1.5	V
V_{OH}	V_{CC} = MIN,	$V_I = 0.5$ V,	$I_{OH} = -0.4$ mA	2.5	3.4		2.7	3.4		V
V_{OL}	V_{CC} = MIN,	$V_I = 1.9$ V	$I_{OL} = 4$ mA		0.25	0.4		0.25	0.4	V
			$I_{OL} = 8$ mA					0.35	0.5	
I_{T+}	$V_{CC} = 5$ V,	$V_I = V_{T+}$			−0.14			−0.14		mA
I_{T-}	$V_{CC} = 5$ V,	$V_I = V_{T-}$			−0.18			−0.18		mA
I_I	V_{CC} = MAX,	$V_I = 7$ V				0.1			0.1	mA
I_{IH}	V_{CC} = MAX,	$V_{IH} = 2.7$ V				20			20	μA
I_{IL}	V_{CC} = MAX,	$V_{IL} = 0.4$ V				−0.4			−0.4	mA
I_{OS}§	V_{CC} = MAX			−20		−100	−20		−100	mA
I_{CCH}	V_{CC} = MAX				8.6	16		8.6	16	mA
I_{CCL}	V_{CC} = MAX				12	21		12	21	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at $V_{CC} = 5$ V, $T_A = 25$ C.
§ Not more than one output should be shorted at a time, and duration of the short-circuit should not exceed one second.

switching characteristics, $V_{CC} = 5$ V, $T_A = 25°C$

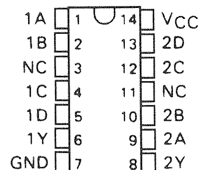
PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	A	Y	$R_L = 2$ kΩ,	$C_L = 15$ pF		15	22	ns
t_{PHL}						15	22	ns

SN54ALS20A, SN54AS20, SN74ALS20A, SN74AS20
DUAL 4-INPUT POSITIVE-NAND GATES

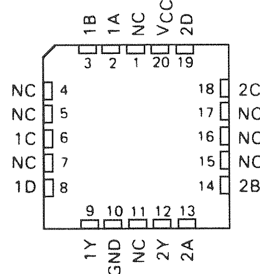
D2661, APRIL 1982—REVISED MAY 1986

- **Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers, and Standard Plastic and Ceramic 300-mil DIPs**
- **Dependable Texas Instruments Quality and Reliability**

SN54ALS20A, SN54AS20 . . . J PACKAGE
SN74ALS20A, SN74AS20 . . . D OR N PACKAGE
(TOP VIEW)



SN54ALS20A, SN54AS20 . . . FK PACKAGE
(TOP VIEW)



NC—No internal connection

description

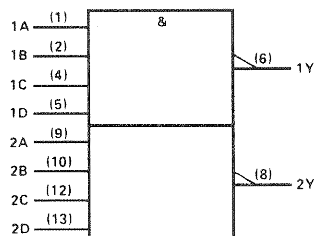
These devices contain two independent 4-input NAND gates. They perform the Boolean functions $Y = \overline{A \cdot B \cdot C \cdot D}$ or $Y = \overline{A} + \overline{B} + \overline{C} + \overline{D}$ in positive logic.

The SN54ALS20A and SN54AS20 are characterized for operation over the full military temperature range of −55°C to 125°C. The SN74ALS20A and SN74AS20 are characterized for operation from 0°C to 70°C.

FUNCTION TABLE (each gate)

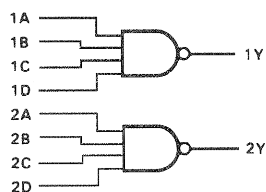
INPUTS				OUTPUT
A	B	C	D	Y
H	H	H	H	L
L	X	X	X	H
X	L	X	X	H
X	X	L	X	H
X	X	X	L	H

logic symbol†



† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.
Pin numbers shown are for D, J, and N packages.

logic diagram (positive logic)



TEXAS INSTRUMENTS
POST OFFICE BOX 655012 • DALLAS, TEXAS 75265

SN54ALS20A, SN74ALS20A
DUAL 4-INPUT POSITIVE-NAND GATES

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} . . . . 7 V
Input voltage . . . . 7 V
Operating free-air temperature range: SN54ALS20A . . . . −55°C to 125°C
SN74ALS20A . . . . 0°C to 70°C
Storage temperature range . . . . −65°C to 150°C

recommended operating conditions

		SN54ALS20A MIN	SN54ALS20A NOM	SN54ALS20A MAX	SN74ALS20A MIN	SN74ALS20A NOM	SN74ALS20A MAX	UNIT
V_{CC}	Supply voltage	4.5	5	5.5	4.5	5	5.5	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.7			0.8	V
I_{OH}	High-level output current			−0.4			−0.4	mA
I_{OL}	Low-level output current			4			8	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating-free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS		SN54ALS20A MIN	SN54ALS20A TYP†	SN54ALS20A MAX	SN74ALS20A MIN	SN74ALS20A TYP†	SN74ALS20A MAX	UNIT
V_{IK}	V_{CC} = 4.5 V,	I_I = −18 mA			−1.5			−1.5	V
V_{OH}	V_{CC} = 4.5 V to 5.5 V,	I_{OH} = −0.4 mA	V_{CC} − 2			V_{CC} − 2			V
V_{OL}	V_{CC} = 4.5 V,	I_{OL} = 4 mA		0.25	0.4		0.25	0.4	V
	V_{CC} = 4.5 V,	I_{OL} = 8 mA					0.35	0.5	
I_I	V_{CC} = 5.5 V,	V_I = 7 V			0.1			0.1	mA
I_{IH}	V_{CC} = 5.5 V,	V_I = 2.7 V			20			20	μA
I_{IL}	V_{CC} = 5.5 V,	V_I = 0.4 V			−0.1			−0.1	mA
I_O‡	V_{CC} = 5.5 V,	V_O = 2.25 V	−30		−112	−30		−112	mA
I_{CCH}	V_{CC} = 5.5 V,	V_I = 0 V		0.22	0.4		0.22	0.4	mA
I_{CCL}	V_{CC} = 5.5 V,	V_I = 4.5 V		0.81	1.5		0.81	1.5	mA

† All typical values are at V_{CC} = 5 V, T_A = 25°C.
‡ The output conditions have been chosen to produce a current that closely approximates one half of the true short-circuit output current, I_{OS}.

switching characteristics (see Note 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC} = 5 V, C_L = 50 pF, R_L = 500 Ω, T_A = 25°C	V_{CC} = 4.5 V to 5.5 V, C_L = 50 pF, R_L = 500 Ω, T_A = MIN to MAX				UNIT
			'ALS20A	SN54ALS20A		SN74ALS20A		
			TYP	MIN	MAX	MIN	MAX	
t_{PLH}	Any	Y	7	1	18	3	11	ns
t_{PHL}	Any	Y	6	1	15	3	10	ns

NOTE 1: Load circuit and voltage waveforms are shown in Section 1.

SN54AS20, SN74AS20
DUAL 4-INPUT POSITIVE-NAND GATES

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} .... 7 V
Input voltage .... 7 V
Operating free-air temperature range: SN54AS20 .... −55°C to 125°C
SN74AS20 .... 0°C to 70°C
Storage temperature range .... −65°C to 150°C

recommended operating conditions

		SN54AS20			SN74AS20			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.5	5	5.5	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			−2			−2	mA
I_{OL}	Low-level output current						20	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating-free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS		SN54AS20			SN74AS20			UNIT
			MIN	TYP†	MAX	MIN	TYP†	MAX	
V_{IK}	V_{CC} = 4.5 V,	I_I = −18 mA			−1.2			−1.2	V
V_{OH}	V_{CC} = 4.5 V to 5.5 V,	I_{OH} = −2 mA	V_{CC} − 2			V_{CC} − 2			V
V_{OL}	V_{CC} = 4.5 V,	I_{OL} = 20 mA		0.35	0.5		0.35	0.5	V
I_I	V_{CC} = 5.5 V,	V_I = 7 V			0.1			0.1	mA
I_{IH}	V_{CC} = 5.5 V,	V_I = 2.7 V			20			20	μA
I_{IL}	V_{CC} = 5.5 V,	V_I = 0.4 V			−0.5			−0.5	mA
I_O‡	V_{CC} = 5.5 V,	V_O = 2.25 V	−30		−112	−30		−112	mA
I_{CCH}	V_{CC} = 5.5 V,	V_I = 0 V		1	1.6		1	1.6	mA
I_{CCL}	V_{CC} = 5.5 V,	V_I = 4.5 V		5.4	8.7		5.4	8.7	mA

† All typical values are at V_{CC} = 5 V, T_A = 25°C.
‡ The output conditions have been chosen to produce a current that closely approximates one half of the true short circuit output current, I_{OS}.

switching characteristics (see Note 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC} = 4.5 V to 5.5 V, C_L = 50 pF, R_L = 500 Ω, T_A = MIN to MAX				UNIT
			SN54AS20		SN74AS20		
			MIN	MAX	MIN	MAX	
t_{PLH}	Any	Y	1	5.5	1	5	ns
t_{PHL}	Any	Y	1	5	1	4.5	ns

NOTE 1: Load circuit and voltage waveforms are shown in Section 1.

TYPES SN54LS20, SN74LS20
DUAL 4-INPUT POSITIVE-NAND GATES

recommended operating conditions

		SN54LS20			SN74LS20			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.7			0.8	V
I_{OH}	High-level output current			– 0.4			– 0.4	mA
I_{OL}	Low-level output current			4			8	mA
T_A	Operating free-air temperature	– 55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS †			SN54LS20			SN74LS20			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN,	I_I = – 18 mA				– 1.5			– 1.5	V
V_{OH}	V_{CC} = MIN,	V_{IL} = MAX,	I_{OH} = – 0.4 mA	2.5	3.4		2.7	3.4		V
V_{OL}	V_{CC} = MIN,	V_{IH} = 2 V,	I_{OL} = 4 mA		0.25	0.4			0.4	V
	V_{CC} = MIN,	V_{IH} = 2 V,	I_{OL} = 8 mA					0.25	0.5	
I_I	V_{CC} = MAX,	V_I = 7 V				0.1			0.1	mA
I_{IH}	V_{CC} = MAX,	V_I = 2.7 V				20			20	μA
I_{IL}	V_{CC} = MAX,	V_I = 0.4 V				– 0.4			– 0.4	mA
I_{OS}§	V_{CC} = MAX			– 20		– 100	– 20		– 100	mA
I_{CCH}	V_{CC} = MAX,	V_I = 0 V			0.4	0.8		0.4	0.8	mA
I_{CCL}	V_{CC} = MAX,	V_I = 4.5 V			1.2	2.2		1.2	2.2	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time, and the duration of the short-circuit should not exceed one second.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 2)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	Any	Y	R_L = 2 kΩ,	C_L = 15 pF		9	15	ns
t_{PHL}						10	15	ns

NOTE 2: See General Information Section for load circuits and voltage waveforms.

HIGH-SPEED CMOS LOGIC

TABLE I
SPECIFICATIONS FOR HC SSI CIRCUITS

D2804, DECEMBER 1982 – REVISED MARCH 1984

absolute maximum ratings over operating free-air temperature range†

Supply voltage range, V_{CC} . . . . −0.5 V to 7 V
Input diode current, I_{IK} (V_I < 0 or V_I > V_{CC}) . . . . ±20 mA
Output diode current, I_{OK} (V_O < 0 or V_O > V_{CC}) . . . . ±20 mA
Continuous output current, I_O (V_O = 0 to V_{CC}) . . . . ±25 mA
Continuous current through V_{CC} or GND pins . . . . ±50 mA
Lead temperature 1,6 mm (1/16 inch) from case for 60 seconds: FH, FK, or J package . . . . 300°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds: FN or N package . . . . 260°C
Storage temperature range . . . . −65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

recommended operating conditions

			SN54HC'			SN74HC'			UNIT
			MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage		2	5	6	2	5	6	V
V_{IH}	High-level input voltage	V_{CC} = 2 V	1.5			1.5			V
		V_{CC} = 4.5 V	3.15			3.15			
		V_{CC} = 6 V	4.2			4.2			
V_{IL}	Low-level input voltage	V_{CC} = 2 V	0		0.3	0		0.3	V
		V_{CC} = 4.5 V	0		0.9	0		0.9	
		V_{CC} = 6 V	0		1.2	0		1.2	
V_I	Input voltage		0		V_{CC}	0		V_{CC}	V
V_O	Output voltage		0		V_{CC}	0		V_{CC}	V
t_t	Input transition (rise and fall) times (except Schmitt-trigger inputs)	V_{CC} = 2 V	0		1000	0		1000	ns
		V_{CC} = 4.5 V	0		500	0		500	
		V_{CC} = 6 V	0		400	0		400	
T_A	Operating free-air temperature		−55		125	−40		85	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS	V_{CC}	T_A = 25°C			SN54HC'		SN74HC'		UNIT
			MIN	TYP	MAX	MIN	MAX	MIN	MAX	
V_{OH} (Totem-pole outputs)	V_I = V_{IH} or V_{IL}, I_{OH} = −20 μA	2 V	1.9	1.998		1.9		1.9		V
		4.5 V	4.4	4.499		4.4		4.4		
		6 V	5.9	5.999		5.9		5.9		
	V_I = V_{IH} or V_{IL}, I_{OH} = −4 mA	4.5 V	3.98	4.30		3.7		3.84		
	V_I = V_{IH} or V_{IL}, I_{OH} = −5.2 mA	6 V	5.48	5.80		5.2		5.34		
I_{OH} (Open-drain outputs)	V_I = V_{IH} or V_{IL}, V_O = V_{CC}	6 V		0.01	0.5		10		5	μA
V_{OL}	V_I = V_{IH} or V_{IL}, I_{OL} = 20 μA	2 V		0.002	0.1		0.1		0.1	V
		4.5 V		0.001	0.1		0.1		0.1	
		6 V		0.001	0.1		0.1		0.1	
	V_I = V_{IH} or V_{IL}, I_{OL} = 4 mA	4.5 V		0.17	0.26		0.4		0.33	
	V_I = V_{IH} or V_{IL}, I_{OL} = 5.2 mA	6 V		0.15	0.26		0.4		0.33	
V_{T+}†		2 V	0.8	1.2	1.5					V
		4.5 V	2	2.5	3.15					
		6 V	2.5	3.3	4.2					
V_{T-}†		2 V	0.3	0.6	0.8					V
		4.5 V	0.9	1.6	2					
		6 V	1.2	2	2.5					
$V_{T+} - V_{T-}$†		2 V	0.2	0.6	1					V
		4.5 V	0.4	0.9	1.4					
		6 V	0.5	1.3	1.7					
I_I	V_I = 0 to V_{CC}	6 V		±0.1	±100		±1000		±1000	nA
I_{CC}	V_I = V_{CC} or 0, I_O = 0	6 V			2		40		20	μA
C_I		2 to 6 V		3	10		10		10	pF

†This parameter applies only for Schmitt-trigger inputs.

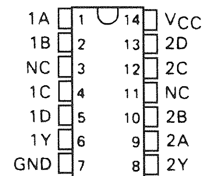
HIGH-SPEED CMOS LOGIC

TYPES SN54HC20, SN74HC20 DUAL 4-INPUT POSITIVE-NAND GATES

D2684, DECEMBER 1982—REVISED MARCH 1984

- Package Options Include Both Plastic and Ceramic Chip Carriers in Addition to Plastic and Ceramic DIPs
- Dependable Texas Instruments Quality and Reliability

SN54HC20 . . . J PACKAGE
SN74HC20 . . . J OR N PACKAGE
(TOP VIEW)

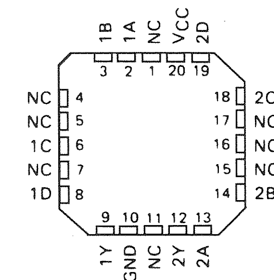


description

These devices contain two independent 4-input NAND gates. They perform the Boolean functions $Y=\overline{A \cdot B \cdot C \cdot D}$ or $Y=\overline{A}+\overline{B}+\overline{C}+\overline{D}$ in positive logic.

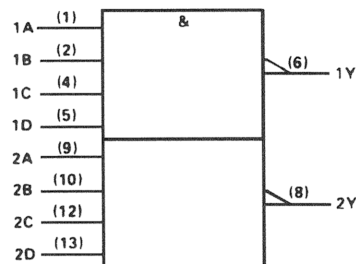
The SN54HC20 is characterized for operation over the full military temperature range of − 55 °C to 125 °C. The SN74HC20 is characterized for operation from − 40 °C to 85 °C.

SN54HC20 . . . FH OR FK PACKAGE
SN74HC20 . . . FH OR FN PACKAGE
(TOP VIEW)



NC—No internal connection

logic symbol



Pin numbers shown are for J and N packages.

FUNCTION TABLE (each gate)

INPUTS				OUTPUT
A	B	C	D	Y
H	H	H	H	L
L	X	X	X	H
X	L	X	X	H
X	X	L	X	H
X	X	X	L	H

maximum ratings, recommended operating conditions, and electrical characteristics

See Table I, page 2-4.

switching characteristics over recommended operating free-air temperature range (unless otherwise noted), C_L = 50 pF (see Note 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC}	T_A = 25°C MIN	T_A = 25°C TYP	T_A = 25°C MAX	SN54HC20 MIN	SN54HC20 MAX	SN74HC20 MIN	SN74HC20 MAX	UNIT
t_{pd}	A, B, C, or D	Y	2 V		45	110		165		140	ns
			4.5 V		14	22		33		28	
			6 V		11	19		28		24	
t_t		Y	2 V		27	75		110		95	ns
			4.5 V		9	15		22		19	
			6 V		7	13		19		16	

C_{pd}	Power dissipation capacitance per gate	No load, T_A = 25°C	25 pF typ

NOTE 1: For load circuit and voltage waveforms, see page 1-14.

SN5432, SN7432
QUADRUPLE 2-INPUT POSITIVE-OR GATES

recommended operating conditions

		SN5432			SN7432			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	Hgh-level input voltage	2			2			V
V_{IL}	Low-level imput voltage			0.8			0.8	V
I_{OH}	High-level output current			− 0.8			− 0.8	mA
I_{OL}	Low-level output current			16			16	mA
T_A	Operating free-air temperature	− 55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†	SN5432			SN7432			UNIT
		MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN, I_I = − 12 mA			− 1.5			− 1.5	V
V_{OH}	V_{CC} = MIN, V_{IH} = 2 V, I_{OH} = − 0.8 mA	2.4	3.4		2.4	3.4		V
V_{OL}	V_{CC} = MIN, V_{IL} = 0.8 V, I_{OL} = 16 mA		0.2	0.4		0.2	0.4	V
I_I	V_{CC} = MAX, V_I = 5.5 V			1			1	mA
I_{IH}	V_{CC} = MAX, V_I = 2.4 V			40			40	μA
I_{IL}	V_{CC} = MAX, V_I = 0.4 V			− 1.6			− 1.6	mA
I_{OS}§	V_{CC} = MAX	− 20		− 55	− 18		− 55	mA
I_{CCH}	V_{CC} = MAX, See Note 2		15	22		15	22	mA
I_{CCL}	V_{CC} = MAX, V_I = 0 V		23	38		23	38	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time.
NOTE 2: One input at 4.5 V, all others at GND.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 3)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Y	R_L = 400 Ω, C_L = 15 pF		10	15	ns
t_{PHL}	A or B	Y	R_L = 400 Ω, C_L = 15 pF		14	22	ns

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

SN54S32, SN74S32
QUADRUPLE 2-INPUT POSITIVE-OR GATES

recommended operating conditions

		SN54S32			SN74S32			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			−1			−1	mA
I_{OL}	Low-level output current			20			20	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS †			SN54S32			SN74S32			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN,	I_I = −18 mA				−1.2			−1.2	V
V_{OH}	V_{CC} = MIN,	V_{IH} = 2 V,	I_{OH} = −1 mA	2.5	3.4		2.7	3.4		V
V_{OL}	V_{CC} = MIN,	V_{IL} = 0.8 V,	I_{OL} = 20 mA			0.5			0.5	V
I_I	V_{CC} = MAX,	V_I = 5.5 V				1			1	mA
I_{IH}	V_{CC} = MAX,	V_I = 2.7 V				50			50	μA
I_{IL}	V_{CC} = MAX,	V_I = 0.5 V				−2			−2	mA
I_{OS}§	V_{CC} = MAX			−40		−100	−40		−100	mA
I_{CCH}	V_{CC} = MAX,	See Note 2			18	32		18	32	mA
I_{CCL}	V_{CC} = MAX,	V_I = 0 V			38	68		38	68	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time and the duration of the short-circuit should not exceed one second.
NOTE 2: One input at 4.5 V, all others at GND.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 3)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Y	R_L = 280 Ω,	C_L = 15 pF		4	7	ns
t_{PHL}						4	7	ns
t_{PLH}	A or B	Y	R_L = 280 Ω,	C_L = 50 pF		5		ns
t_{PHL}						5		ns

NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

SN54LS37, SN74LS37
QUADRUPLE 2-INPUT POSITIVE-NAND BUFFERS

recommended operating conditions

		SN54LS37			SN74LS37			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.7			0.8	V
I_{OH}	High-level output current			−1.2			−1.2	mA
I_{OL}	Low-level output current			12			24	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS †	SN54LS37 MIN	SN54LS37 TYP‡	SN54LS37 MAX	SN74LS37 MIN	SN74LS37 TYP‡	SN74LS37 MAX	UNIT
V_{IK}	V_{CC} = MIN, I_I = −18 mA			−1.5			−1.5	V
V_{OH}	V_{CC} = MIN, V_{IL} = MAX, I_{OH} = −1.2 mA	2.5	3.4		2.7	3.4		V
V_{OL}	V_{CC} = MIN, V_{IH} = 2 V, I_{OL} = 12 mA		0.25	0.4		0.25	0.4	V
	V_{CC} = MIN, V_{IH} = 2 V, I_{OL} = 24 mA					0.35	0.5	
I_I	V_{CC} = MAX, V_I = 7 V			0.1			0.1	mA
I_{IH}	V_{CC} = MAX, V_I = 2.7 V			20			20	μA
I_{IL}	V_{CC} = MAX, V_I = 0.4 V			−0.4			−0.4	mA
I_{OS}§	V_{CC} = MAX	−30		−130	−30		−130	mA
I_{CCH}	V_{CC} = MAX, V_I = 0 V		0.9	2		0.9	2	mA
I_{CCL}	V_{CC} = MAX, V_I = 4.5 V		6	12		6	12	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time, and the duration of the short-circuit should not exceed one second.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see note 2)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Y	R_L = 667 Ω, C_L = 45 pF		12	24	ns
t_{PHL}					12	24	ns

NOTE 2: Load circuits and voltage waveforms are shown in Section 1.

SN5486, SN7486
QUADRUPLE 2-INPUT EXCLUSIVE-OR GATES

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1) . . . 7 V
Input voltage . . . 5.5 V
Operating free-air temperature range: SN5486 . . . −55°C to 125°C
SN7486 . . . 0°C to 70°C
Storage temperature range . . . −65°C to 150°C

NOTE 1: Voltage values are with respect to network ground terminal.

recommended operating conditions

	SN5486			SN7486			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{CC}	4.5	5	5.5	4.75	5	5.25	V
High-level output current, I_{OH}			−800			−800	μA
Low-level output current, I_{OL}			16			16	mA
Operating free-air temperature, T_A	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS†	SN5486			SN7486			UNIT
			MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IH}	High-level input voltage		2			2			V
V_{IL}	Low-level input voltage				0.8			0.8	V
V_{IK}	Input clamp voltage	V_{CC} = MIN, I_I = −8 mA			−1.5			−1.5	V
V_{OH}	High-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OH} = −800 μA	2.4	3.4		2.4	3.4		V
V_{OL}	Low-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OL} = 16 mA		0.2	0.4		0.2	0.4	V
I_I	Input current at maximum input voltage	V_{CC} = MAX, V_I = 5.5 V			1			1	mA
I_{IH}	High-level input current	V_{CC} = MAX, V_I = 2.4 V			40			40	μA
I_{IL}	Low-level input current	V_{CC} = MAX, V_I = 0.4 V			−1.6			−1.6	mA
I_{OS}	Short-circuit output current§	V_{CC} = MAX	−20		−55	−18		−55	mA
I_{CC}	Supply current	V_{CC} = MAX, See Note 2		30	43		30	50	mA

†For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable type.
‡All typical values are at V_{CC} = 5 V, T_A = 25°C.
§Not more than one output should be shorted at a time.
NOTE 2: I_{CC} is measured with the inputs grounded and the outputs open.

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER¶	FROM (INPUT)	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Other input low	C_L = 15 pF, R_L = 400 Ω, See Note 3		15	23	ns
t_{PHL}					11	17	
t_{PLH}	A or B	Other input high			18	30	ns
t_{PHL}					13	22	

¶t_{PLH} = propagation delay time, low-to-high-level output
t_{PHL} = propagation delay time, high-to-low-level output
NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

SN54LS86A, SN74LS86A
QUADRUPLE 2-INPUT EXCLUSIVE-OR GATES

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1) . . . 7 V
Input voltage . . . 7 V
Operating free-air temperature range: SN54LS86A . . . −55°C to 125°C
SN74LS86A . . . 0°C to 70°C
Storage temperature range . . . −65°C to 150°C

NOTE 1: Voltage values are with respect to network ground terminal.

recommended operating conditions

	SN54LS86A			SN74LS86A			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{CC}	4.5	5	5.5	4.75	5	5.25	V
High-level output current, I_{OH}			−400			−400	μA
Low-level output current, I_{OL}			4			8	mA
Operating free-air temperature, T_A	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†		SN54LS86A			SN74LS86A			UNIT
			MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IH} High-level input voltage			2			2			V
V_{IL} Low-level input voltage					0.7			0.8	V
V_{IK} Input clamp voltage	V_{CC} = MIN, I_I = −18 mA				−1.5			−1.5	V
V_{OH} High-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = V_{IL} max, I_{OH} = −400 μA		2.5	3.4		2.7	3.4		V
V_{OL} Low-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = V_{IL}max	I_{OL} = 4 mA		0.25	0.4		0.25	0.4	V
		I_{OL} = 8 mA					0.35	0.5	
I_I Input current at maximum input voltage	V_{CC} = MAX, V_I = 7 V				0.2			0.2	mA
I_{IH} High-level input current	V_{CC} = MAX, V_I = 2.7 V				40			40	μA
I_{IL} Low-level input current	V_{CC} = MAX, V_I = 0.4 V				−0.8			−0.8	mA
I_{OS} Short-circuit output current§	V_{CC} = MAX		−20		−100	−20		−100	mA
I_{CC} Supply current	V_{CC} = MAX, See Note 2			6.1	10		6.1	10	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable type.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C
§ Not more than one output should be shorted at a time.
NOTE 2: I_{CC} is measured with the inputs grounded and the outputs open.

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER¶	FROM (INPUT)	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Other input low	C_L = 15 pF, R_L = 2 kΩ, See Note 3		12	23	ns
t_{PHL}					10	17	
t_{PLH}	A or B	Other input high			20	30	ns
t_{PHL}					13	22	

¶ t_{PLH} = propagation delay time, low-to-high-level output
t_{PHL} = propagation delay time, high-to-low-level output
NOTE 3: Load circuits and voltage waveforms are shown in Section 1.

SN54ALS86, SN54AS86A, SN74ALS86, SN74AS86A
QUADRUPLE 2-INPUT EXCLUSIVE-OR GATES

SDAS006B – APRIL 1982 – REVISED DECEMBER 1994

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{CC} .... 7 V
Input voltage, V_I .... 7 V
Operating free-air temperature range, T_A: SN54ALS86 .... −55°C to 125°C
SN74ALS86 .... 0°C to 70°C
Storage temperature range .... −65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

recommended operating conditions

		SN54ALS86			SN74ALS86			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.5	5	5.5	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.7			0.8	V
I_{OH}	High-level output current			−0.4			−0.4	mA
I_{OL}	Low-level output current			4				mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS		SN54ALS86			SN74ALS86			UNIT
			MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = 4.5 V,	I_I = −18 mA			−1.5			−1.5	V
V_{OH}	V_{CC} = 4.5 V to 5.5 V,	I_{OH} = −0.4 mA	V_{CC} −2			V_{CC} −2			V
V_{OL}	V_{CC} = 4.5 V	I_{OL} = 4 mA		0.25	0.4		0.25	0.4	V
		I_{OL} = 8 mA					0.35	0.5	
I_I	V_{CC} = 5.5 V,	V_I = 7 V			0.1			0.1	mA
I_{IH}	V_{CC} = 5.5 V,	V_I = 2.7 V			20			20	μA
I_{IL}	V_{CC} = 5.5 V,	V_I = 0.4 V			−0.1			−0.1	mA
I_O§	V_{CC} = 5.5 V,	V_O = 2.25 V	−20		−112	−30		−112	mA
I_{CC}	V_{CC} = 5.5 V,	All inputs at 4.5 V		3.9	5.9		3.9	5.9	mA

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ The output conditions have been chosen to produce a current that closely approximates one half of the true short-circuit output current, I_{OS}.

switching characteristics (see Figure 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC} = 4.5 V to 5.5 V, C_L = 50 pF, R_L = 500 Ω, T_A = MIN to MAX¶				UNIT
			SN54ALS86		SN74ALS86		
			MIN	MAX	MIN	MAX	
t_{PLH}	A or B (other input low)	Y	3	22	3	17	ns
t_{PHL}			2	14	2	12	
t_{PLH}	A or B (other input high)	Y	3	22	3	17	ns
t_{PHL}			2	12	2	10	

¶ For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.

SN54ALS86, SN54AS86A, SN74ALS86, SN74AS86A
QUADRUPLE 2-INPUT EXCLUSIVE-OR GATES

SDAS006B – APRIL 1982 – REVISED DECEMBER 1994

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{CC} .... 7 V
Input voltage, V_I .... 7 V
Operating free-air temperature range, T_A: SN54AS86A .... −55°C to 125°C
SN74AS86A .... 0°C to 70°C
Storage temperature range .... −65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

recommended operating conditions

		SN54AS86A			SN74AS86A			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.5	5	5.5	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			−2			−2	mA
I_{OL}	Low-level output current			20			20	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS		SN54AS86A			SN74AS86A			UNIT
			MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = 4.5 V,	I_I = −18 mA			−1.2			−1.2	V
V_{OH}	V_{CC} = 4.5 V to 5.5 V,	I_{OH} = −2 mA	V_{CC} −2			V_{CC} −2			V
V_{OL}	V_{CC} = 4.5 V,	I_{OL} = 20 mA		0.35	0.5		0.35	0.5	V
I_I	V_{CC} = 5.5 V,	V_I = 7 V			0.1			0.1	mA
I_{IH}	V_{CC} = 5.5 V,	V_I = 2.7 V			20			20	μA
I_{IL}	V_{CC} = 5.5 V,	V_I = 0.4 V			−0.5			−0.5	mA
I_O§	V_{CC} = 5.5 V,	V_O = 2.25 V	−30		−112	−30		−112	mA
I_{CCH}	V_{CC} = 5.5 V,	$V_{I(A)}$ = 4.5 V, $V_{I(B)}$ = 0		11	18		11	18	mA
I_{CCL}	V_{CC} = 5.5 V,	V_I = 4.5 V		20	38		20	38	mA

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ The output conditions have been chosen to produce a current that closely approximates one half of the true short-circuit output current, I_{OS}.

switching characteristics (see Figure 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC} = 4.5 V to 5.5 V, C_L = 50 pF, R_L = 500 Ω, T_A = MIN to MAX¶				UNIT
			SN54AS86A		SN74AS86A		
			MIN	MAX	MIN	MAX	
t_{PLH}	A or B (other input low)	Y	2	8.5	2	7.5	ns
t_{PHL}			2	8	2	6.5	
t_{PLH}	A or B (other input high)	Y	1	8	1	6.5	ns
t_{PHL}			1	9	1	7	

¶ For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.



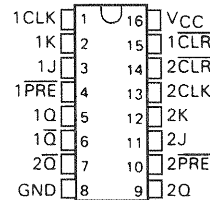
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

SN54LS112A, SN54S112, SN74LS112A, SN74S112A DUAL J-K NEGATIVE-EDGE-TRIGGERED FLIP-FLOPS WITH PRESET AND CLEAR

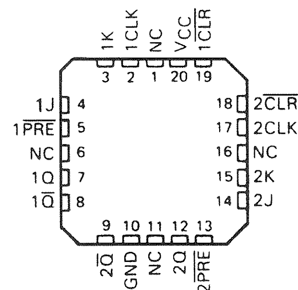
D2661, APRIL 1982 – REVISED MARCH 1988

- Fully Buffered to Offer Maximum Isolation from External Disturbance
- Package Options Include Plastic "Small Outline" Packages, Ceramic Chip Carriers and Flat Packages, and Plastic and Ceramic DIPs
- Dependable Texas Instruments Quality and Reliability

SN54LS112A, SN54S112 . . . J OR W PACKAGE
SN74LS112A, SN74S112A . . . D OR N PACKAGE
(TOP VIEW)



SN54LS112A, SN54S112 . . . FK PACKAGE
(TOP VIEW)



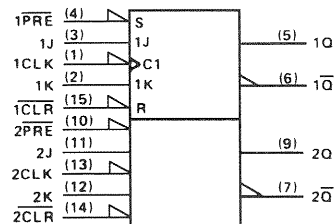
NC – No internal connection

description

These devices contain two independent J-K negative-edge-triggered flip-flops. A low level at the preset and clear inputs sets or resets the outputs regardless of the levels of the other inputs. When preset and clear are inactive (high), data at the J and K inputs meeting the setup time requirements are transferred to the outputs on the negative-going edge of the clock pulse. Clock triggering occurs at a voltage level and is not directly related to the rise time of the clock pulse. Following the hold time interval, data at the J and K inputs may be changed without affecting the levels at the outputs. These versatile flip-flops can perform as toggle flip-flops by tying J and K high.

The SN54LS112A and SN54S112 are characterized for operation over the full military temperature range of −55°C to 125°C. The SN74LS112A and SN74S112A are characterized for operation from 0°C to 70°C.

logic symbol‡



‡This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.
Pin numbers shown are for D, J, N, and W packages.

FUNCTION TABLE (each flip-flop)

INPUTS					OUTPUTS	
PRE	CLR	CLK	J	K	Q	Q̄
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H†	H†
H	H	↓	L	L	Q_0	$\overline{Q}_0$
H	H	↓	H	L	H	L
H	H	↓	L	H	L	H
H	H	↓	H	H	TOGGLE	
H	H	H	X	X	Q_0	$\overline{Q}_0$

† The output levels in this configuration are not guaranteed to meet the minimum levels for V_{OH} if the lows at preset and clear are near V_{IL} minimum. Furthermore, this configuration is nonstable; that is, it will not persist when either preset or clear returns to its inactive (high) level.

SN54LS112A, SN74LS112A DUAL J-K NEGATIVE-EDGE-TRIGGERED FLIP-FLOPS WITH PRESET AND CLEAR

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS†	SN54LS112A MIN	SN54LS112A TYP‡	SN54LS112A MAX	SN74LS112A MIN	SN74LS112A TYP‡	SN74LS112A MAX	UNIT
V_{IK}		V_{CC} = MIN, I_I = −18 mA			−1.5			−1.5	V
V_{OH}		V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = MAX, I_{OH} = −0.4 mA	2.5	3.4		2.7	3.4		V
V_{OL}		V_{CC} = MIN, V_{IL} = MAX, V_{IH} = 2 V, I_{OL} = 4 mA		0.25	0.4		0.25	0.4	V
		V_{CC} = MIN, V_{IL} = MAX, V_{IH} = 2 V, I_{OL} = 8 mA					0.35	0.5	
I_I	J or K	V_{CC} = MAX, V_I = 7 V			0.1			0.1	mA
	$\overline{CLR}$ or $\overline{PRE}$				0.3			0.3	
	CLK				0.4			0.4	
I_{IH}	J or K	V_{CC} = MAX, V_I = 2.7 V			20			20	μA
	$\overline{CLR}$ or $\overline{PRE}$				60			60	
	CLK				80			80	
I_{IL}	J or K	V_{CC} = MAX, V_I = 0.4 V			−0.4			−0.4	mA
	All other				−0.8			−0.8	
I_{OS}§		V_{CC} = MAX, see Note 2	−20		−100	−20		−100	mA
I_{CC} (Total)		V_{CC} = MAX, see Note 3		4	6		4	6	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25 °C.
§ Not more than one output should be shorted at a time, and the duration of the short-circuit should not exceed one second.
NOTES: 2. For certain devices where state commutation can be caused by shorting an output to ground, an equivalent test may be performed with V_O = 2.25 V and 2.125 V for the '54 family and the '74 family, respectively, with the minimum and maximum limits reduced to one half of their stated values.
3. With all outputs open, I_{CC} is measured with the Q and $\overline{Q}$ outputs high in turn. At the time of measurement, the clock input is grounded.

switching characteristics, V_{CC} = 5 V, T_A = 25 °C (see Note 4)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
f_{max}			R_L = 2 kΩ, C_L = 15 pF	30	45		MHz
t_{PLH}	$\overline{CLR}$, $\overline{PRE}$ or CLK	Q or $\overline{Q}$			15	20	ns
t_{PHL}					15	20	ns

NOTE 4: Load circuits and voltage waveforms are shown in Section 1.

recommended operating conditions

			SN54LS112A MIN	SN54LS112A NOM	SN54LS112A MAX	SN74LS112A MIN	SN74LS112A NOM	SN74LS112A MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage		2			2			V
V_{IL}	Low-level input voltage				0.7			0.8	V
I_{OH}	High-level output current				−0.4			−0.4	mA
I_{OL}	Low-level output current				4			8	mA
f_{clock}	Clock frequency		0		30	0		30	MHz
t_w	Pulse duration	CLK high	20			20			ns
		$\overline{PRE}$ or $\overline{CLR}$ low	25			25			
t_{su}	Set up time-before CLK↓	Data high or low	20			20			ns
		$\overline{CLR}$ inactive	25			25			
		$\overline{PRE}$ inactive	20			20			
t_h	Hold time-data after CLK↓		0			0			ns
T_A	Operating free-air temperature		−55		125	0		70	°C

SN54S112, SN74S112A
DUAL J-K NEGATIVE-EDGE-TRIGGERED
FLIP-FLOPS WITH PRESET AND CLEAR

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS†	SN54S112			SN74S112A			UNIT
			MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}		V_{CC} = MIN, I_I = −18 mA			−1.2			−1.2	V
V_{OH}		V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = MAX, I_{OH} = −1 mA	2.5	3.4		2.7	3.4		V
V_{OL}		V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = 0.8 V, I_{OL} = 20 mA			0.5			0.5	V
I_I		V_{CC} = MAX, V_I = 5.5 V			1			1	mA
I_{IH}	J or K	V_{CC} = MAX, V_I = 2.7 V			50			50	μA
	All other				100			100	
I_{IL}	J or K	V_{CC} = MAX, V_I = 0.5 V			−1.6			−1.6	mA
	$\overline{CLR}$§				−7			−7	
	$\overline{PRE}$§				−7			−7	
	CLK				−4			−4	
I_{OS}¶		V_{CC} = MAX	−40		−100	−40		−100	mA
I_{CC}#		V_{CC} = MAX, see Note 3		15	25		15	25	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Clear is tested with preset high and preset is tested with clear high.
¶ Not more than one output should be shorted at a time, and the duration of the short-circuit should not exceed one second.
Values are average per flip-flop.
NOTE 3: With all outputs open, I_{CC} is measured with the Q and $\overline{Q}$ outputs high in turn. At the time of measurement, the clock input is grounded.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 4)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
f_{max}			R_L = 280 Ω, C_L = 15 pF	80	125		MHz
t_{PLH}	$\overline{PRE}$ or $\overline{CLR}$	Q or $\overline{Q}$			4	7	ns
t_{PHL}	$\overline{PRE}$ or $\overline{CLR}$ (CLK high)	$\overline{Q}$ or Q			5	7	ns
	$\overline{PRE}$ or $\overline{CLR}$ (CLK low)				5	7	
t_{PLH}	CLK	Q or $\overline{Q}$			4	7	ns
t_{PHL}					5	7	ns

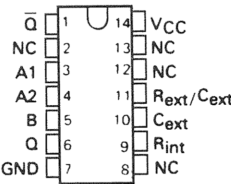
NOTE 4: Load circuits and voltage waveforms are shown in Section 1.

SN54121, SN74121 MONOSTABLE MULTIVIBRATORS WITH SCHMITT-TRIGGER INPUTS

MAY 1983 – REVISED MARCH 1988

- Programmable Output Pulse Width
  With R_{int} . . . 35 ns Typ
  With R_{ext}/C_{ext} . . . 40 ns to 28 Seconds
- Internal Compensation for Virtual Temperature Independence
- Jitter-Free Operation up to 90% Duty Cycle
- Inhibit Capability

SN54121 . . . J OR W PACKAGE
SN74121 . . . N PACKAGE
(TOP VIEW)

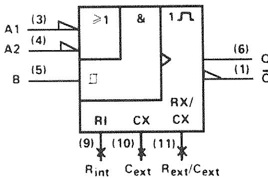


NC - No internal connection.

FUNCTION TABLE

INPUTS			OUTPUTS	
A1	A2	B	Q	Q̄
L	X	H	L	H
X	L	H	L†	H†
X	X	L	L†	H†
H	H	X	L†	H†
H	↓	H	⊓	⊔
↓	H	H	⊓	⊔
↓	↓	H	⊓	⊔
L	X	↑	⊓	⊔
X	L	↑	⊓	⊔

logic symbol‡



‡This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

For explanation of function table symbols, see page

† These lines of the function table assume that the indicated steady state conditions at the A and B inputs have been setup long enough to complete any pulse started before the setup.

description

These multivibrators feature dual negative-transition-triggered inputs and a single positive-transition-triggered input which can be used as an inhibit input. Complementary output pulses are provided.

Pulse triggering occurs at a particular voltage level and is not directly related to the transition time of the input pulse. Schmitt-trigger input circuitry (TTL hysteresis) for the B input allows jitter-free triggering from inputs with transition rates as slow as 1 volt/second, providing the circuit with an excellent noise immunity of typically 1.2 volts. A high immunity to V_{CC} noise of typically 1.5 volts is also provided by internal latching circuitry.

Once fired, the outputs are independent of further transitions of the inputs and are a function only of the timing components. Input pulses may be of any duration relative to the output pulse. Output pulse length may be varied from 40 nanoseconds to 28 seconds by choosing appropriate timing components. With no external timing components (i.e., R_{int} connected to V_{CC}, C_{ext} and R_{ext}/C_{ext} open), an output pulse of typically 30 or 35 nanoseconds is achieved which may be used as a d-c triggered reset signal. Output rise and fall times are TTL compatible and independent of pulse length.

Pulse width stability is achieved through internal compensation and is virtually independent of V_{CC} and temperature. In most applications, pulse stability will only be limited by the accuracy of external timing components.

Jitter-free operation is maintained over the full temperature and V_{CC} ranges for more than six decades of timing capacitance (10 pF to 10 μF) and more than one decade of timing resistance (2 kΩ to 30 kΩ for the SN54121 and 2 kΩ to 40 kΩ for the SN74121). Throughout these ranges, pulse width is defined by the relationship $t_{w(out)} = C_{ext}R_T\ln 2 \approx 0.7\ C_{ext}R_T$. In circuits where pulse cutoff is not critical, timing capacitance up to 1000 μF and timing resistance as low as 1.4 kΩ may be used. Also, the range of jitter-free output pulse widths is extended if V_{CC} is held to 5 volts and free-air temperature is 25°C. Duty cycles as high as 90% are achieved when using maximum recommended R_T. Higher duty cycles are available if a certain amount of pulse-width jitter is allowed.



SN54121, SN74121
MONOSTABLE MULTIVIBRATORS
WITH SCHMITT-TRIGGER INPUTS

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS†		MIN	TYP‡	MAX	UNIT
V_{IH}	High-level input voltage at B input	V_{CC} = MIN		2			V
V_{IL}	Low-level input voltage at A input	V_{CC} = MIN				0.8	V
V_{T+}	Positive-going threshold voltage at B input	V_{CC} = MIN			1.55	2	V
V_{T-}	Negative-going threshold voltage at B input	V_{CC} = MIN		0.8	1.35		V
V_{IK}	Input clamp voltage	V_{CC} = MIN,	I_I = −12 mA			−1.5	V
V_{OH}	High-level output voltage	V_{CC} = MIN,	I_{OH} = MAX	2.4	3.4		V
V_{OL}	Low-level output voltage	V_{CC} = MIN,	I_{OL} = MAX		0.2	0.4	V
I_I	Input current at maximum input voltage	V_{CC} = MAX,	V_I = 5.5 V			1	mA
I_{IH}	High-level input current	V_{CC} = MAX, V_I = 2.4 V	A1 or A2			40	μA
			B			80	
I_{IL}	Low-level input current	V_{CC} = MAX, V_I = 0.4 V	A1 or A2			−1.6	mA
			B			−3.2	
I_{OS}	Short-circuit output current§	V_{CC} = MAX	54 Family	−20		−55	mA
			74 Family	−18		−55	
I_{CC}	Supply current	V_{CC} = MAX	Quiescent		13	25	mA
			Triggered		23	40	

†For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡All typical values are at V_{CC} = 5 V, T_A = 25°C.
§Not more than one output should be shorted at a time.

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER		TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	Propagation delay time, low-to-high-level Q output from either A input	C_L = 15 pF, R_L = 400 Ω, See Note 4	C_{ext} = 80 pF, R_{int} to V_{CC}		45	70	ns
t_{PLH}	Propagation delay time, low-to-high-level Q output from B input				35	55	ns
t_{PHL}	Propagation delay time, high-to-low level $\overline{Q}$ output from either A input				50	80	ns
t_{PHL}	Propagation delay time, high-to-low level $\overline{Q}$ output from B input				40	65	ns
$t_{w(out)}$	Pulse width obtained using internal timing resistor		C_{ext} = 80 pF, R_{int} to V_{CC}	70	110	150	ns
$t_{w(out)}$	Pulse width obtained with zero timing capacitance		C_{ext} = 0, R_{int} to V_{CC}		30	50	ns
$t_{w(out)}$	Pulse width obtained using external timing resistor		C_{ext} = 100 pF, R_T = 10 kΩ	600	700	800	ns
			C_{ext} = 1 μF, R_T = 10 kΩ	6	7	8	ms

NOTE 4: Load circuits and voltage waveforms are shown in Section 1.

SN54LS192, SN54LS193, SN74LS192, SN74LS193
SYNCHRONOUS 4-BIT UP/DOWN COUNTERS (DUAL CLOCK WITH CLEAR)

recommended operating conditions

		SN54LS192 SN54LS193			SN74LS192 SN74LS193			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
I_{OH}	High-level output current			−400			−400	μA
I_{OL}	Low-level output current			4			8	mA
f_{clock}	Clock frequency	0		25	0		25	MHz
t_w	Width of any input pulse	20			20			ns
t_{su}	Clear inactive-state setup time	15			15			ns
t_{su}	Load inactive-state setup time	15			15			ns
t_{su}	Data setup time (see Figure 1)	20			20			ns
t_h	Data hold time	5			5			ns
T_A	Operating free-air temperature range	−55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS†		SN54LS192 SN54LS193			SN74LS192 SN74LS193			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IH}	High-level input voltage			2			2			V
V_{IL}	Low-level input voltage					0.7			0.8	V
V_{IK}	Input clamp voltage	V_{CC} = MIN, I_I = −18 mA				−1.5			−1.5	V
V_{OH}	High-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = V_{IL} max, I_{OH} = −400 μA		2.5	3.4		2.7	3.4		V
V_{OL}	Low-level output voltage	V_{CC} = MIN, V_{IH} = 2 V, V_{IL} = V_{IL} max	I_{OL} = 4 mA		0.25	0.4		0.15	0.4	V
			I_{OL} = 8 mA					0.35	0.5	
I_I	Input current at maximum input voltage	V_{CC} = MAX, V_I = 7 V				0.1			0.1	mA
I_{IH}	High-level input current	V_{CC} = MAX, V_I = 2.7 V				20			20	μA
I_{IL}	Low-level input current	V_{CC} = MAX, V_I = 0.4 V				−0.4			−0.4	mA
I_{OS}	Short-circuit output current§	V_{CC} = MAX		−20		−100	−20		−100	mA
I_{CC}	Supply current	V_{CC} = MAX, See Note 2			19	34		19	34	mA

†For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions for the applicable type.
‡All typical values are at V_{CC} = 5 V, T_A = 25°C.
§Not more than one output should be shorted at a time, and duration of the short-circuit should not exceed one second.
NOTE 2: I_{CC} is measured with all outputs open, clear and load inputs grounded, and all other inputs at 4.5 V.

switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS	MIN	TYP	MAX	UNIT
f_{max}			C_L = 15 pF, R_L = 2 kΩ, See Figures 1 and 2	25	32		MHz
t_{PLH}	UP	$\overline{CO}$			17	26	ns
t_{PHL}	UP	$\overline{CO}$			18	24	ns
t_{PLH}	DOWN	$\overline{BO}$			16	24	ns
t_{PHL}	DOWN	$\overline{BO}$			15	24	ns
t_{PLH}	UP OR DOWN	Q			27	38	ns
t_{PHL}	UP OR DOWN	Q			30	47	ns
t_{PLH}	$\overline{LOAD}$	Q			24	40	ns
t_{PHL}	$\overline{LOAD}$	Q			25	40	ns
t_{PHL}	CLR	Q			23	35	ns

SN54ALS193, SN74ALS193A
SYNCHRONOUS 4-BIT UP/DOWN BINARY COUNTERS WITH DUAL CLOCK AND CLEAR

SDAS211B – DECEMBER 1982 – REVISED DECEMBER 1994

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{CC} .... 7 V
Input voltage, V_I .... 7 V
Operating free-air temperature range, T_A: SN54ALS193 .... –55°C to 125°C
SN74ALS193A .... 0°C to 70°C
Storage temperature range .... –65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

recommended operating conditions

			SN54ALS193			SN74ALS193A			UNIT
			MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage		4.5	5	5.5	4.5	5	5.5	V
V_{IH}	High-level input voltage		2			2			V
V_{IL}	Low-level input voltage				0.7			0.8	V
I_{OH}	High-level output current				–0.4			–0.4	mA
I_{OL}	Low-level output current				4			8	mA
f_{clock}	Clock frequency		0		20	0		30	MHz
t_w	Pulse duration	CLR high	10			10			ns
		$\overline{\text{LOAD}}$ low	25			20			
		UP or DOWN high or low	30			16.5			
t_{su}	Setup time	Data before $\overline{\text{LOAD}}$↑	25			20			ns
		CLR inactive before UP or DOWN	20			20			
		$\overline{\text{LOAD}}$ inactive before UP or DOWN	20			20			
t_h	Hold time	Data after $\overline{\text{LOAD}}$↑	5			5			ns
		UP high after DOWN↑	0			0			
		DOWN high after UP↑	0			0			
T_A	Operating free-air temperature		–55		125	0		70	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS		SN54ALS193			SN74ALS193A			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}		V_{CC} = 4.5 V,	I_I = –18 mA			–1.5			–1.5	V
V_{OH}		V_{CC} = 4.5 V to 5.5 V,	I_{OH} = –0.4 mA	V_{CC} – 2			V_{CC} – 2			V
V_{OL}		V_{CC} = 4.5 V	I_{OL} = 4 mA		0.25	0.4		0.25	0.4	V
			I_{OL} = 8 mA					0.35	0.5	
I_I		V_{CC} = 5.5 V,	V_I = 7 V			0.1		0.35	0.1	mA
I_{IH}		V_{CC} = 5.5 V,	V_I = 2.7 V			20			20	µA
I_{IL}	UP or DOWN	V_{CC} = 5.5 V,	V_I = 0.4 V			–0.2			–0.2	mA
	All others					–0.1			–0.1	
I_O§		V_{CC} = 5.5 V,	V_O = 2.25 V	–20		–112	–30		– 112	mA
I_{CC}		V_{CC} = 5.5 V,	See Note 1		12	22		12	22	mA

‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ The output conditions have been chosen to produce a current that closely approximates one half of the true short-circuit output current, I_{OS}.
NOTE 1: I_{CC} is measured with the clear and load inputs grounded and all other inputs at 4.5 V.



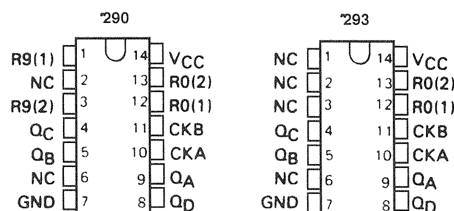
SN54290, SN54293, SN54LS290, SN54LS293, SN74290, SN74293, SN74LS290, SN74LS293
DECADE AND 4-BIT BINARY COUNTERS

MARCH 1974 – REVISED MARCH 1988

'290, 'LS290 . . . DECADE COUNTERS
'293, 'LS293 . . . 4-BIT BINARY COUNTERS

- GND and V_{CC} on Corner Pins (Pins 7 and 14 Respectively)

SN54290, SN54LS290, SN54293, SN54LS293 . . . J OR W PACKAGE
SN74290, SN74293 . . . N PACKAGE
SN74LS290, SN74LS293 . . . D OR N PACKAGE
(TOP VIEW)



description

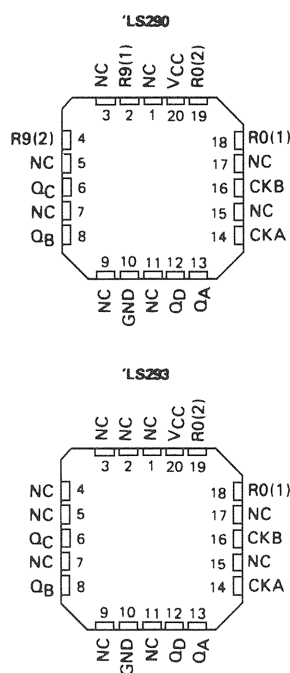
The SN54290/SN74290, SN54LS290/SN74LS290, SN54293/SN74293, and SN54LS293/SN74LS293 counters are electrically and functionally identical to the SN5490A/SN7490A, SN54LS90/SN74LS90, SN5493A/SN7493A, and SN54LS93/SN74LS93, respectively. Only the arrangement of the terminals has been changed for the '290, 'LS290, '293, and 'LS293.

Each of these monolithic counters contains four master-slave flip-flops and additional gating to provide a divide-by-two counter and a three-stage binary counter for which the count cycle length is divide-by-five for the '290 and 'LS290 and divide-by-eight for the '293 and 'LS293.

All of these counters have a gated zero reset and the '290 and 'LS290 also have gated set-to-nine inputs for use in BCD nine's complement applications.

To use the maximum count length (decade or four-bit binary) of these counters, the B input is connected to the Q_A output. The input count pulses are applied to input A and the outputs are as described in the appropriate function table. A symmetrical divide-by-ten count can be obtained from the '290 and 'LS290 counters by connecting the Q_D output to the A input and applying the input count to the B input which gives a divide-by-ten square wave at output Q_A.

SN54LS290, SN54LS293 . . . FK PACKAGE
(TOP VIEW)



NC - No internal connection

SN54290, SN54293, SN54LS290, SN54LS293, SN74290, SN74293, SN74LS290, SN74LS293 DECADE AND 4-BIT BINARY COUNTERS

'290, 'LS290
BCD COUNT SEQUENCE
(See Note A)

COUNT	OUTPUT			
	Q_D	Q_C	Q_B	Q_A
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
	H	L	L	H

'290, 'LS290
BI-QUINARY (5-2)
(See Note B)

COUNT	OUTPUT			
	Q_A	Q_D	Q_C	Q_B
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	H	L	L	L
6	H	L	L	H
7	H	L	H	L
8	H	L	H	H
9	H	H	L	L

'290, 'LS290
RESET/COUNT FUNCTION TABLE

RESET INPUTS				OUTPUT			
$R_{0(1)}$	$R_{0(2)}$	$R_{9(1)}$	$R_{9(2)}$	Q_D	Q_C	Q_B	Q_A
H	H	L	X	L	L	L	L
H	H	X	L	L	L	L	L
X	X	H	H	H	L	L	H
X	L	X	L	COUNT			
L	X	L	X	COUNT			
L	X	X	L	COUNT			
X	L	L	X	COUNT			

'293, 'LS293
RESET/COUNT FUNCTION TABLE

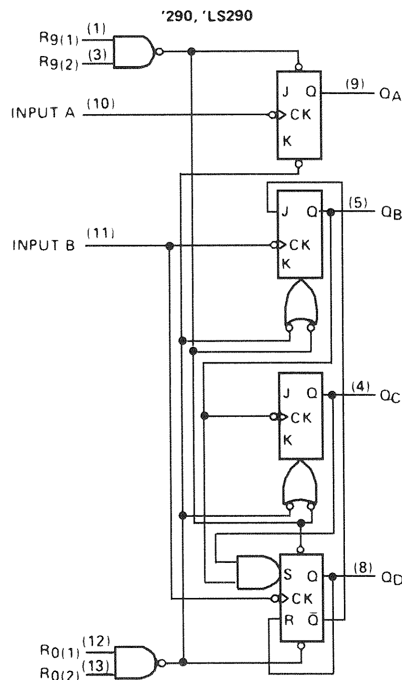
RESET INPUTS		OUTPUT			
$R_{0(1)}$	$R_{0(2)}$	Q_D	Q_C	Q_B	Q_A
H	H	L	L	L	L
L	X	COUNT			
X	L	COUNT			

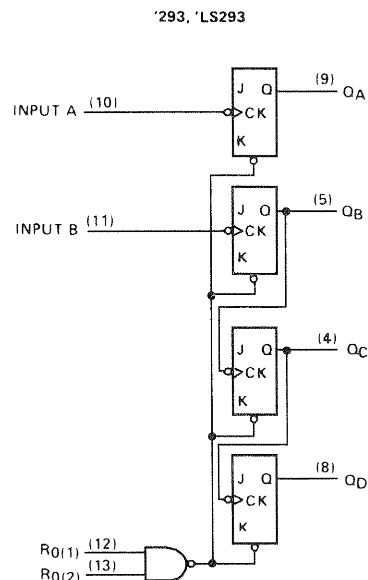
'293, 'LS293
COUNT SEQUENCE
(See Note C)

COUNT	OUTPUT			
	Q_D	Q_C	Q_B	Q_A
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
9	H	L	L	H
10	H	L	H	L
11	H	L	H	H
12	H	H	L	L
13	H	H	L	H
14	H	H	H	L
15	H	H	H	H

NOTES: A. Output Q_A is connected to input B for BCD count.
B. Output Q_D is connected to input A for bi-quinary count.
C. Output Q_A is connected to input B.
D. H = high level, L = low level, X = irrelevant

logic diagrams (positive logic)





Pin numbers shown are for D, J, N, and W packages.
The J and K inputs shown without connection are for reference only and are functionally at a high level.

4001B 4002B

QUAD 2-INPUT NOR GATE • DUAL 4-INPUT NOR GATE

DESCRIPTION – These CMOS logic elements provide the positive input NOR function. The outputs are fully buffered for highest noise immunity and pattern insensitivity of output impedance.

4001B
LOGIC AND CONNECTION DIAGRAM
DIP (TOP VIEW)

NOTE:
The Flatpak versions have the same pinouts (Connection Diagram) as the Dual In-line Package.

4002B
LOGIC AND CONNECTION DIAGRAM
DIP (TOP VIEW)

DC CHARACTERISTICS: V_{DD} as shown, V_{SS} = 0 V

SYMBOL	PARAMETER		LIMITS									UNITS	TEMP	TEST CONDITIONS See Note 1
			V_{DD} = 5 V			V_{DD} = 10 V			V_{DD} = 15 V					
			MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX			
I_{DD}	Quiescent Power Supply Current	XC			1			2			4	µA	MIN, 25°C	All inputs at 0 V or V_{DD}
					7.5			15			30		MAX	
		XM			0.25			0.5			1	µA	MIN, 25°C	
					7.5			15			30		MAX	

AC CHARACTERISTICS: V_{DD} as shown, V_{SS} = 0 V, T_A = 25°C, 4001B only (See Note 2)

SYMBOL	PARAMETER	LIMITS									UNITS	TEST CONDITIONS See Note 2
		V_{DD} = 5 V			V_{DD} = 10 V			V_{DD} = 15V				
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX		
t_{PLH}	Propagation Delay		60	110		25	60		20	48	ns	C_L = 50 pF,
t_{PHL}			60	110		25	60		20	48	ns	R_L = 200 kΩ
t_{TLH}	Output Transition Time		60	135		30	70		20	45	ns	Input Transition
t_{THL}			60	135		30	70		20	45	ns	Times ≤ 20 ns

AC CHARACTERISTICS: V_{DD} as shown, V_{SS} = 0 V, T_A = 25°C, 4002B only

SYMBOL	PARAMETER	LIMITS									UNITS	TEST CONDITIONS See Note 2
		V_{DD} = 5 V			V_{DD} = 10 V			V_{DD} = 15V				
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX		
t_{PLH}	Propagation Delay		65	110		30	60		20	48	ns	C_L = 50 pF,
t_{PHL}			70	110		30	60		23	48	ns	R_L = 200 kΩ
t_{TLH}	Output Transition Time		75	135		40	70		30	45	ns	Input Transition
t_{THL}			60	135		23	70		15	45	ns	Times ≤ 20 ns

NOTES:
1. Additional DC Characteristics are listed in this section under 4000B Series CMOS Family Characteristics.
2. Propagation Delays and Output Transition Times are graphically described in this section under 4000B Series CMOS Family Characteristics

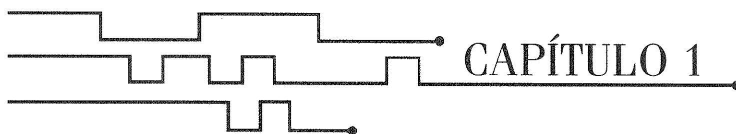
4001B

FAIRCHILD 4000B SERIES CMOS FAMILY CHARACTERISTICS

DC CHARACTERISTICS FOR THE 4000B SERIES CMOS FAMILY – Parametric Limits listed below are guaranteed for the entire Fairchild CMOS Family unless otherwise specified on the individual data sheets.

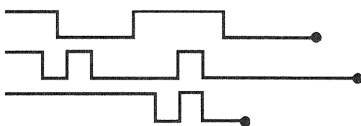
DC CHARACTERISTICS: V_{DD} = 5 V, V_{SS} = 0 V

SYMBOL	PARAMETER	LIMITS MIN	TYP	MAX	UNITS	TEMP	TEST CONDITIONS
V_{IH}	Input HIGH Voltage	3.5			V	All	Guaranteed Input HIGH Voltage
V_{IL}	Input LOW Voltage			1.5	V	All	Guaranteed Input Low Voltage
V_{OH}	Output HIGH Voltage	4.95			V	Min, 25°C	I_{OH} < 1 µA, Inputs at 0 or 5 V per the Logic Function or Truth Table
		4.95				MAX	
		4.5			V	All	I_{OH} < 1 µA, Inputs at 1.5 or 3.5 V
V_{OL}	Output LOW Voltage			0.05	V	MIN, 25°C	I_{OL} < 1 µA, Inputs at 0 or 5 V per the Logic Function or Truth Table
				0.05		MAX	
				0.5	V	All	I_{OL} < 1 µA, Inputs at 1.5 or 3.5 V
I_{OH}	Output HIGH Current	-0.63			mA	MIN, 25°C	V_{OUT} = 4.6 V; Inputs at 0 or 5 V per the Logic Function or Truth Table
		-0.36				MAX	
I_{OL}	Output LOW Current	1			mA	MIN,	V_{OUT} = 0.4 V; Inputs at 0 or 5 V per the Logic Function or Truth Table
		0.8				25°C	
		0.4				MAX	
C_{IN}	Input Capacitance Per Unit Load			7.5	pF	25°C	Any Input
I_{DD}	Quiescent Power Supply Current — Gates — XC			1	µA	MIN, 25°C	All Inputs at 0 V or V_{DD} for all Valid Input Combinations
				7.5		MAX	
	Gates — XM			0.25	µA	MIN, 25°C	
				7.5		MAX	
	Buffers and Flip-Flops — XC			4	µA	MIN, 25°C	
				30		MAX	
	Buffers and Flip-Flops — XM			1	µA	MIN, 25°C	
				30		MAX	
	MSI — XC			20	µA	MIN, 25°C	
				150		MAX	
	MSI — XM			5	µA	MIN, 25°C	
				150		MAX	



CAPÍTULO 1

Conceitos Introdutórios



SUMÁRIO

OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Distinguir entre representações analógicas e digitais.
- Relacionar as vantagens e desvantagens das técnicas digitais quando comparadas com as analógicas.
- Compreender a necessidade de utilização de conversores analógico-digitais (conversores A/D) e conversores digital-analógicos (conversores D/A).
- Converter números decimais em binários e vice-versa.
- Identificar sinais digitais típicos.
- Relacionar as diversas tecnologias de fabricação de circuitos integrados.
- Identificar um diagrama de tempo.
- Enumerar as diferenças entre transmissão paralela e serial.
- Descrever a propriedade de memória.
- Descrever as principais partes de um computador digital e compreender suas funções.
- Fazer a distinção entre microcomputadores, microprocessadores e microcontroladores.

INTRODUÇÃO

No mundo atual, o termo digital tornou-se parte do nosso vocabulário no dia-a-dia por causa da maneira profunda pela qual os circuitos e as técnicas digitais tornaram-se amplamente utilizados em quase todas as áreas de nossas vidas, como: computadores, automação, robôs, medicina, transportes, entretenimento, exploração do espaço etc. Você está prestes a começar uma excitante jornada, na qual descobrirá os princípios fundamentais, os conceitos e as operações que são comuns a todos os sistemas digitais, desde a mais simples chave liga-desliga até o mais complexo computador. Se este livro for bem-sucedido, você adquirirá um profundo conhecimento de como os sistemas digitais funcionam e deverá estar apto a aplicar este conhecimento na análise e manutenção de qualquer sistema digital.

Vamos começar introduzindo alguns conceitos básicos que são a parte vital da tecnologia digital; estes conceitos serão complementados mais adiante, à medida que se tornar necessário. Também introduziremos alguns termos, importantes para iniciarmos o estudo nesta nova área de conhecimento, assim como acrescentaremos, a cada capítulo, novos termos àqueles já estudados.

1-1 REPRESENTAÇÕES NUMÉRICAS

Na ciência, na tecnologia, nos negócios e na verdade em qualquer outro campo, estamos constantemente lidando com *quantidades*. Quantidades são medidas, monitoradas, gravadas, manipuladas aritmeticamente, observadas, ou de algum outro modo utilizadas na maioria dos sistemas físicos. É importante que ao lidarmos com diversas quantidades sejamos capazes de representar seus valores de modo eficiente e exato. Existem basicamente duas formas de representar o valor numérico de quantidades: a **analógica** e a **digital.**

Representações Analógicas

Na **representação analógica,** o valor de uma quantidade é proporcional ao valor de uma tensão ou corrente, ou ainda de uma medida de movimento. Um exemplo disso é o velocímetro de um automóvel, no qual a deflexão do ponteiro é proporcional à velocidade do automóvel. A posição angular do ponteiro indica o valor da velocidade do automóvel, inclusive acompanhando qualquer mudança que ocorrer na velocidade do automóvel ao ser acelerado ou freado.

Um outro exemplo é o termostato utilizado para controlar a temperatura de uma sala, no qual a curvatura de uma lâmina bimetálica é proporcional à temperatura ambiente. À medida que a temperatura na sala se altera, a curvatura da lâmina também se altera proporcionalmente.

Ainda um outro exemplo de representação analógica pode ser encontrado no conhecido microfone de áudio. Neste dispositivo, a tensão de saída gerada é proporcional à amplitude das ondas sonoras que atingem o microfone. As variações na tensão de saída acompanham as mesmas variações das ondas sonoras na entrada.

Quantidades representadas na forma analógica tais como aquelas citadas anteriormente possuem uma importante característica: *elas podem variar em um determinado intervalo contínuo de valores.* A velocidade de um automóvel pode assumir qualquer valor no intervalo entre zero e, digamos, 160 km/h. De modo similar, a tensão de saída de um microfone pode estar em qualquer ponto de um intervalo de zero a 10 mV (por exemplo: 1 mV, 2,3724 mV, 9,9999 mV).

Representações Digitais

Na **representação digital,** as quantidades são representadas não por outras quantidades proporcionais, mas por símbolos chamados *dígitos.* Por exemplo, um relógio digital, que fornece as horas do dia na forma de dígitos decimais que representam as horas, os minutos (e às vezes segundos). Como sabemos, as horas do dia mudam continuamente, mas a leitura do relógio digital não varia continuamente; em vez disso, ela varia em passos de um minuto (ou um segundo). Em outras palavras, esta forma de representação digital das horas do dia varia em passos *discretos,* quando comparada com a representação fornecida por um relógio analógico, em que as mudanças no mostrador ocorrem de modo contínuo.

A diferença principal entre as formas de representação analógica e digital pode então ser simplesmente simbolizada da seguinte maneira:

analógica ≡ contínua
digital ≡ discreta (passo a passo)

Por causa da natureza discreta da representação digital, não existe ambigüidade na leitura de uma quantidade representada nesta forma, enquanto na representação analógica a leitura é geralmente sujeita a interpretação.

EXEMPLO 1-1

Quais dos itens a seguir referem-se à forma de representação digital e quais se referem à analógica?

(a) Chave de dez posições
(b) A corrente elétrica na tomada na parede
(c) A temperatura de uma sala
(d) Grãos de areia na praia
(e) Velocímetro de automóvel

Solução

(a) Digital
(b) Analógica
(c) Analógica
(d) Digital, uma vez que o número de grãos pode assumir apenas um determinado número de valores discretos (inteiros) e não qualquer valor possível dentro de um intervalo contínuo.
(e) Analógica, se o velocímetro for do tipo de ponteiro; digital se possuir um mostrador numérico.

Questões de Revisão*

1. Descreva, de modo resumido, a principal diferença entre as formas de representação digital e analógica.

1-2 SISTEMAS DIGITAIS E ANALÓGICOS

Um **sistema digital** é uma combinação de dispositivos projetados para lidar com informações lógicas ou com quantidades físicas representadas de forma digital, isto é, estas quantidades só podem assumir valores discretos. Estes dispositivos são geralmente eletrônicos, mas também podem ser mecânicos, magnéticos ou pneumáticos. Dentre os sistemas digitais mais comuns podemos citar computadores e calculadoras digitais, equipamento de aúdio e vídeo digital e o sistema telefônico — o maior sistema digital no mundo.

Um **sistema analógico** contém dispositivos que podem manipular quantidades físicas que são representadas de forma analógica. Em um sistema analógico, as quantidades físicas podem variar sobre um intervalo contínuo de valores. Por exemplo: a amplitude do sinal de saída de um receptor de rádio pode ter qualquer valor entre zero e o limite máximo. Outros sistemas analógicos bastante comuns são os amplificadores de aúdio, equipamento de gravação e reprodução de fita magnética, e um simples interruptor do tipo *dimmer*.

*As respostas das questões de revisão podem ser encontradas no fim de cada capítulo.

Vantagens das Técnicas Digitais

Uma crescente maioria das aplicações na eletrônica, bem como em muitas outras áreas, utiliza técnicas digitais para realizar operações anteriormente realizadas através de métodos analógicos. As principais razões da mudança para técnicas digitais são:

1. *Sistemas digitais geralmente são mais fáceis de projetar.* Isto se deve ao fato de que os circuitos utilizados são *circuitos de chaveamento*, em que os valores *exatos* de tensão ou corrente não são importantes, mas apenas o intervalo (ALTO ou BAIXO) no qual eles se localizam.
2. *Fácil armazenamento de informação.* Isto é alcançado por circuitos de chaveamento especiais, capazes de capturar a informação e guardá-la pelo tempo que for necessário.
3. *Maior exatidão e precisão.* Sistemas digitais podem manipular quantos dígitos de precisão forem necessários, para o que basta adicionar um número maior de circuitos de chaveamento. Em sistemas analógicos, a precisão está geralmente limitada a três ou quatro dígitos, porque os valores de corrente e tensão são diretamente dependentes dos valores dos componentes dos circuitos e também são afetados por flutuações randômicas (ruído).
4. *A operação do sistema pode ser programada.* É bastante simples projetar sistemas digitais cuja operação pode ser controlada por um conjunto de instruções, constituindo um *programa*. À medida que a tecnologia avança, a programação de sistemas vem se tornando cada vez mais simples. Sistemas analógicos também podem ser *programados*; entretanto, a variedade e a complexidade das operações disponíveis são bastante limitadas.
5. *Circuitos digitais são menos afetados pelo ruído.* Flutuações espúrias na tensão (ruído) não são tão críticas em sistemas digitais porque o valor exato da tensão não é tão importante, desde que a amplitude do ruído também não seja tão grande que nos impeça de distinguir corretamente os níveis lógicos.
6. *Um maior número de circuitos digitais pode ser colocado em um circuito integrado.* É verdade que circuitos analógicos também foram beneficiados com o grande desenvolvimento da tecnologia de fabricação de circuitos integrados, mas sua complexidade e a utilização de componentes economicamente inviáveis de serem integrados (capacitores de alto valor, resistores de precisão, indutores, transformadores) têm impedido que sistemas analógicos alcancem o mesmo nível de integração.

Limitações das Técnicas Digitais

Realmente existe apenas uma única desvantagem quando utilizamos técnicas digitais:

O mundo real é quase totalmente analógico.

A maioria das quantidades físicas é originalmente analógica, e elas são freqüentemente as entradas e saídas monitoradas, operadas e controladas por um sistema. Alguns exemplos são temperatura, pressão, posição, velocidade, nível de líquidos, vazão etc. Estamos habituados a expressar estas quantidades *digitalmente*, de modo que quando dizemos que a temperatura é de 64° (63,8° se quisermos ser mais

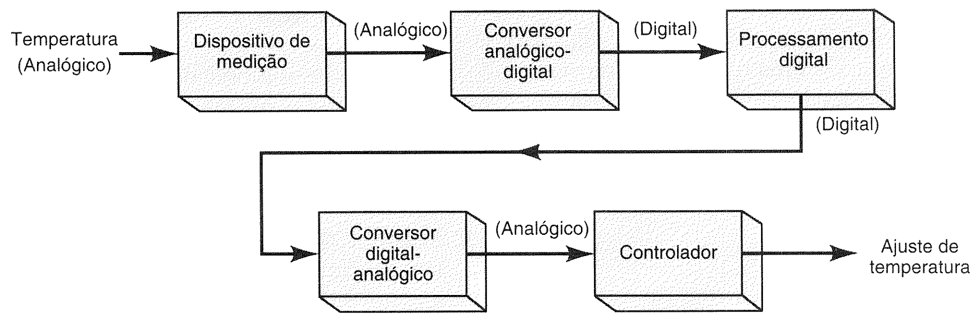


Fig. 1-1 Diagrama de blocos de um sistema de controle de temperatura que utiliza técnicas de processamento digital, possíveis graças às conversões analógico-digitais.

precisos) estamos na verdade fazendo uma aproximação digital de uma grandeza inerentemente analógica.

Para tirar proveito das técnicas digitais quando estivermos lidando com entradas e saídas analógicas, três passos devem ser seguidos:

1. Converter as entradas analógicas para a forma digital.
2. Processar a informação digital.
3. Converter as saídas digitais de volta à forma analógica.

A Fig. 1-1 mostra um diagrama de blocos de um típico sistema de controle de temperatura. Como se pode ver no diagrama, a temperatura é medida por um dispositivo analógico e o valor medido é então convertido para uma representação na forma digital por um **conversor analógico-digital (conversor A/D).** Esta é, então, processada por um circuito digital, que pode incluir ou não um computador digital. A saída digital é então convertida de volta à forma analógica por um **conversor digital-analógico (conversor D/A).** Esta saída analógica é fornecida como entrada a um controlador que realiza algum tipo de ação para ajustar a temperatura.

Um outro bom exemplo de conversão entre representações nas formas analógicas e digitais é a que se verifica na gravação de áudio. Compact disks (CDs) se sobressaíram e tomaram conta da indústria fonográfica por oferecerem melhores meios de gravação e reprodução de música. O processo de produção e reprodução em CD pode ser descrito genericamente como se segue: (1) os sons dos instrumentos e das vozes produzem uma tensão analógica no microfone; (2) este sinal analógico é convertido em um formato digital, usando um processo de conversão de analógico para digital; (3) a informação digital é armazenada na superfície do CD; (4) durante a reprodução, o CD player coleta a informação digital da superfície do CD e a converte em um sinal analógico, que é amplificado e enviado aos alto-falantes de onde pode ser captado pelo ouvido.

A necessidade de conversão entre formas analógicas e digitais pode ser considerada uma desvantagem por causa do seu custo e complexidade adicionais. Um outro fator que geralmente é importante é o tempo extra necessário para realizar estas conversões. Em muitas aplicações, esses fatores são compensados pelas numerosas vantagens de usarmos técnicas digitais, em razão das quais a conversão entre quantidades digitais e analógicas tornou-se algo bastante comum na tecnologia atual.

Existem situações, entretanto, em que a utilização de técnicas analógicas é mais simples e econômica. Por exemplo: a amplificação de sinais é mais facilmente realizada com o auxílio de circuitos analógicos.

É comum observar as técnicas analógicas e digitais serem utilizadas em um mesmo sistema de modo a se tirar proveito das vantagens de cada uma das técnicas. Nesses sistemas *híbridos*, uma das mais importantes etapas do projeto é determinar em que partes devem ser empregadas as técnicas analógicas e aquelas em que devem ser utilizadas técnicas digitais.

O Futuro É Digital

É bastante seguro prever que a maioria dos futuros avanços em muitas (senão em todas) áreas da tecnologia será realizada no domínio digital. O ritmo acelerado desses avanços pode inclusive exceder o crescimento fenomenal que tem sido verificado nos últimos anos — um período em que vimos: 35% das famílias americanas e 50% dos jovens na faixa dos 13 aos 19 anos com computadores pessoais em casa; aproximadamente 30 milhões de pessoas na Internet; 90% de todos os computadores pessoais vendidos em 1995 possuíam modems e unidades de CD-ROM; automóveis com 50 microprocessadores; microprocessadores presentes nas coisas mais comuns, desde torradeiras, termostatos e cartões de cumprimentos até secretárias eletrônicas, videocassetes e máquinas de lavar. E o futuro nos promete ainda mais. No início do século vinte e um, suas abotoaduras ou seus brincos poderão se comunicar com outros, através de satélites, e terão mais poder computacional do que o computador que você possui hoje em casa ou no escritório. Telefones serão capazes de receber, ordenar e talvez até responder chamadas, como um secretária bem-treinada. Na escola, crianças serão capazes de colher idéias e informações e entrar em contato com outras crianças de todo o mundo. Quando você assistir televisão por uma hora, o que estará vendo foi transmitido para sua casa em menos de um segundo e armazenado na memória do computador presente na sua TV, para ser visto de acordo com a sua conveniência. Ler sobre um lugar a 8.000 km de distância pode incluir uma experiência sensorial de estar lá. E isto é apenas a ponta do *iceberg*.*

*As informações e previsões futurísticas citadas aqui foram extraídas do livro *Being Digital*, de Nicholas Negroponte, Vintage Books, 1995, pp. 5-7.

Em outras palavras, a tecnologia digital vai continuar a invadir rapidamente o cotidiano de nossas vidas, bem como vai alcançar novas fronteiras que talvez não tenhamos nem sequer imaginado. Tudo o que podemos fazer é aprender o máximo possível sobre esta tecnologia, agüentar firme e aproveitar a viagem.

Questões de Revisão

1. Quais são as vantagens das técnicas digitais sobre as analógicas?
2. Qual é a maior limitação que existe para se usar técnicas digitais?

1-3 SISTEMAS DE NUMERAÇÃO DIGITAL

Muitos sistemas de numeração são usados na tecnologia digital. Os mais comuns são o decimal, o binário, o octal e o hexadecimal. O sistema decimal é naturalmente o sistema mais familiar para todos, uma vez que ele é uma ferramenta que utilizamos todos os dias. Examinar algumas de suas características nos ajudará a obter uma melhor compreensão dos outros sistemas.

Sistema Decimal

O **sistema decimal** é composto de *10* algarismos ou símbolos. Estes 10 símbolos são 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9. Utilizando estes símbolos como *dígitos* de um número, podemos expressar qualquer quantidade. O sistema decimal é também chamado de sistema de *base 10* porque possui 10 dígitos e evoluiu naturalmente do fato de que as pessoas têm 10 dedos. De fato, a palavra "dígito" é derivada da palavra latina usada para denominar "dedo".

O sistema decimal é um sistema de *valor posicional*, isto é, um sistema no qual o valor do dígito depende de sua posição. Por exemplo: considere o número decimal 453. Sabemos que o dígito 4, na verdade, representa 4 *centenas*; o 5 representa 5 *dezenas* e o 3 representa 3 *unidades*. Em essência, o 4 possui o maior peso dos três dígitos; a ele nos referimos como *dígito mais significativo* (*MSD — Most Significant Digit*). O 3 possui o menor peso e é chamado de *dígito menos significativo* (*LSD — Least Significant Digit*).

Considere um outro exemplo, 27,35. Este número é na verdade igual a 2 dezenas mais 7 unidades mais 3 décimos mais 5 centésimos, ou $2 \times 10 + 7 \times 1 + 3 \times 0,1 + 5 \times 0,01$. A vírgula decimal é usada para separar a parte inteira da parte fracionária do número.

De modo mais rigoroso, as várias posições relativas à vírgula decimal possuem pesos que podem ser expressos em potências de 10. Isto pode ser visto na Fig. 1-2, onde o número 2745,214 está representado. A vírgula decimal separa as potências positivas de 10 daquelas que são negativas. O número 2745,214 é então igual a

$$(2 \times 10^{+3}) + (7 \times 10^{+2}) + (4 \times 10^{1}) + (5 \times 10^{0}) + (2 \times 10^{-1}) + (1 \times 10^{-2}) + (4 \times 10^{-3})$$

Em geral, qualquer número é simplesmente a soma dos produtos de cada valor do dígito pelo seu peso devido à sua posição.

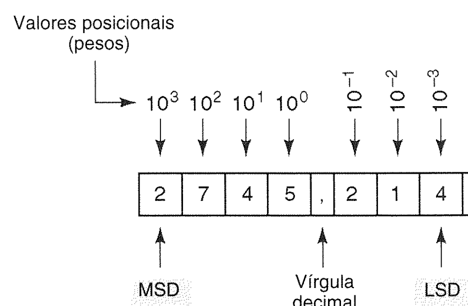


Fig. 1-2 Valores posicionais do sistema de numeração decimal são potências de 10.

Contagem Decimal

Quando fazemos uma contagem no sistema decimal, começamos com 0 na posição das unidades e vamos tomando cada símbolo em progressão até atingirmos 9. Quando isto acontece, adicionamos 1 à posição de maior peso e mais próxima e começamos de novo com o 0, na primeira posição (veja a Fig. 1-3). Este processo continua até que a contagem de 99 seja alcançada. Neste momento, adicionamos 1 à terceira posição e começamos de novo com 0 nas duas primeiras posições. Este procedimento pode ser seguido continuamente, qualquer que seja o número que desejemos contar.

É importante notar que, na contagem decimal, a posição correspondente às unidades (LSD) troca de valor a cada passo da contagem; a posição correspondente às dezenas muda a cada 10 passos da contagem, a posição correspondente às centenas muda a cada 100 passos da contagem e assim por diante.

Outra característica do sistema decimal é que, se utilizarmos dois dígitos, podemos contar até $10^2 = 100$ números diferentes (0 a 99);* utilizando 3 dígitos, podemos contar

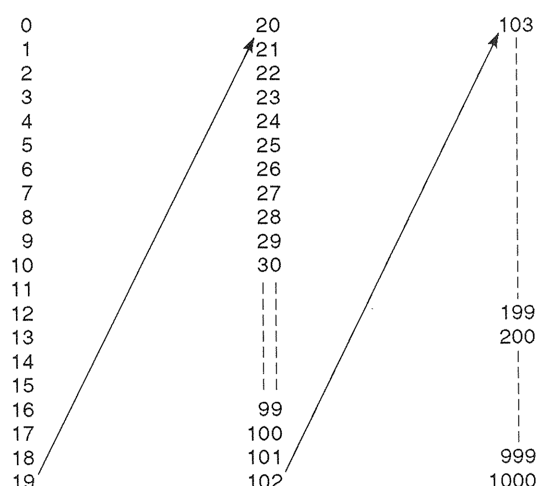


Fig. 1-3 Contagem decimal.

*Zero é contado como um número.

até 1000 números (0 a 999), e assim sucessivamente. De modo geral, com *N* dígitos, podemos contar até 10^N números distintos, começando do zero e incluindo-o na contagem. O maior número possível será sempre igual a $10^N - 1$.

Sistema Binário

Infelizmente, o sistema decimal não se presta para ser implementado satisfatoriamente em sistemas digitais. Por exemplo: é bastante difícil projetar um equipamento eletrônico que possa trabalhar com 10 níveis diferentes de tensão (um para cada algarismo decimal, do 0 ao 9). Por outro lado, é muito fácil implementar circuitos eletrônicos simples e precisos que operem somente com dois níveis de tensão. Por esta razão, quase todo sistema digital usa o sistema de numeração binário (base 2) como sistema de numeração básico para suas operações, embora outros sistemas de numeração, às vezes, sejam usados em conjunção com o sistema binário.

No **sistema binário** existem apenas dois símbolos ou valores possíveis para os dígitos, 0 e 1. Ainda assim, este sistema de base 2 pode ser usado para representar qualquer valor que possa ser representado no sistema decimal ou em qualquer outro sistema. Entretanto, de um modo geral, ele utilizará um número maior de dígitos binários para expressar um dado valor.

Todas as afirmações feitas anteriormente em relação ao sistema decimal são aplicáveis do mesmo modo ao sistema binário. O sistema binário também é um sistema de valor posicional, onde cada dígito binário possui seu próprio valor ou peso expresso como uma potência de dois. Isto é ilustrado na Fig. 1-4.

Na figura, as posições à esquerda da *vírgula binária* (contrapartida da vírgula decimal) representam as potências positivas de 2, e as posições à direita representam as potências negativas de 2. O número 1011,101 é mostrado na figura. Para encontrar o seu equivalente no sistema decimal, simplesmente fazemos a soma dos produtos de cada dígito (0 ou 1) pelo seu respectivo peso.

$$
\begin{aligned}
1011{,}101_2 &= (1\times 2^3) + (0\times 2^2) + (1\times 2^1) + (1\times 2^0) \\
&\quad + (1\times 2^{-1}) + (0\times 2^{-2}) + (1\times 2^{-3}) \\
&= 8 + 0 + 2 + 1 + 0{,}5 + 0 + 0{,}125 \\
&= 11{,}625_{10}
\end{aligned}
$$

Observe que na operação anterior os índices (2 e 10) são usados para indicar a base na qual o número em questão está expresso. Esta convenção é usada para evitar confusão sempre que mais de um sistema de numeração estiver sendo empregado.

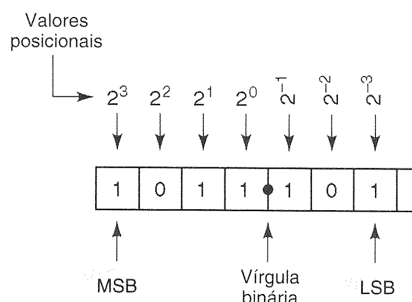


Fig. 1-4 Valores posicionais do sistema de numeração binário são potências de 2.

No sistema binário, o termo *dígito binário* é geralmente abreviado para **bit** (*binary digit*), que usaremos daqui por diante. Assim, para o número mostrado na Fig. 1-4 existem quatro dígitos à esquerda da vírgula binária, representando a parte inteira do número, e três bits à direita, representando a parte fracionária. O bit mais significativo (MSB — *Most Significant Bit*) é o bit mais à esquerda (o de maior peso). O bit menos significativo (LSB — *Least Significant Bit*) é o bit mais à direita (o de menor peso). Estes bits estão indicados na Fig. 1-4.

Contagem Binária

Quando lidamos com números binários, geralmente iremos nos restringir a um número específico de bits. Esta restrição é baseada no conjunto de circuitos que está sendo usado para representar estes números binários. Vamos usar números de 4 bits para ilustrar o método para a contagem em binário.

A seqüência (mostrada na Fig. 1-5) começa com todos os bits em 0; é a chamada *contagem zero*. Para cada contagem sucessiva, a posição referente às unidades (2^0) *comuta*, isto é, ela troca o seu valor binário pelo outro. Cada vez que o bit das unidades trocar de 1 para 0, a posição de peso dois (2^1) vai comutar (trocar de estado). Cada vez que o bit da posição de peso dois mudar de 1 para 0, o bit da posição de peso quatro (2^2) vai comutar (mudar de estado). Do mesmo modo, cada vez que o bit da posição de peso quatro mudar de 1 para 0, o bit da posição de peso oito (2^3) comuta (muda de estado). Este processo continuaria para os bits de mais alta ordem, se o número binário tivesse mais do que quatro bits.

A seqüência de contagem binária tem outra importante característica, como está mostrado na Fig. 1-5. O bit das unidades (LSB) muda de 0 para 1 ou de 1 para 0 a *cada* contagem. O segundo bit (posição de peso dois) fica em 0 por duas contagens e depois em 1 por duas contagens, e

Pesos →	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$	Decimal equivalente
	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	2
	0	0	1	1	3
	0	1	0	0	4
	0	1	0	1	5
	0	1	1	0	6
	0	1	1	1	7
	1	0	0	0	8
	1	0	0	1	9
	1	0	1	0	10
	1	0	1	1	11
	1	1	0	0	12
	1	1	0	1	13
	1	1	1	0	14
	1	1	1	1	15
				↑ LSB	

Fig. 1-5 Seqüência de contagem binária.

depois em 0 por mais duas contagens e assim sucessivamente. O terceiro bit fica em 0 por quatro contagens e depois fica em 1 por quatro contagens e assim sucessivamente. O quarto bit (posição de peso oito) fica em 0 por oito contagens e depois fica em 1 por oito contagens. Se desejássemos continuar a contagem, adicionaríamos mais bits, e este padrão continuaria com grupos de 0s e 1s alternando-se em grupos de 2^{N-1}. Por exemplo: usando o quinto bit, teríamos este alternando dezesseis 0s com dezesseis 1s e assim por diante.

Como vimos no sistema decimal, também é verdade para o sistema binário que usando N bits possamos contar até 2^N contagens. Por exemplo: com dois bits, podemos contar até $2^2 = 4$ contagens (00_2 até 11_2); com quatro bits, podemos contar até $2^4 = 16$ contagens (0000_2 até 1111_2), e assim por diante. A última contagem sempre terá todos os bits iguais a 1 e será igual a $2^N - 1$ no sistema decimal. Por exemplo: usando quatro bits, a última contagem será $1111_2 = 2^4 - 1 = 15_{10}$.

EXEMPLO 1-2

Qual é o maior número que pode ser representado usando oito bits?

Solução

$$2^N - 1 = 2^8 - 1 = 255_{10} = 11111111_2.$$

Esta foi uma breve introdução ao sistema de numeração binário e sua relação com o sistema decimal. Vamos despender muito mais tempo nestes dois sistemas e em vários outros no próximo capítulo.

Questões de Revisão

1. Qual é o número decimal equivalente a 1101011_2?
2. Qual é o próximo número binário que se segue a 10111_2 na seqüência de contagem?
3. Qual é o maior valor decimal que pode ser representado usando-se 12 bits?

1-4 REPRESENTAÇÃO DE QUANTIDADES BINÁRIAS

Em sistemas digitais, a informação que está sendo processada geralmente se apresenta sob forma binária. Quantidades binárias podem ser representadas por qualquer dispositivo que apresente apenas dois estados de operação ou condições possíveis. Por exemplo: uma chave possui apenas dois estados: aberta ou fechada. Podemos arbitrar que uma chave aberta represente o dígito binário 0, ou simplesmente binário 0, e a chave fechada represente o binário 1. A partir destas indicações, podemos representar qualquer número binário como está mostrado na Fig. 1-6(a), em que os estados das várias chaves representam 10010_2.

Um outro exemplo é mostrado na Fig. 1-6(b), em que as perfurações no papel são usadas para representar números binários. Um furo representa o binário 1, e a ausência de furo representa o binário 0.

Existem vários outros dispositivos que possuem apenas dois estados de operação, ou que podem ser operados apenas em condições extremas. Entre estes podemos citar: lâmpada elétrica (acesa ou escura), diodo (conduzindo ou não conduzindo), relé (energizado ou desenergizado), transistor (cortado ou saturado), célula fotoelétrica (iluminada ou escura), termostato (aberto ou fechado), embreagem mecânica (engrenada ou desengrenada) e um ponto específico de um disco magnético (magnetizado ou desmagnetizado).

Em sistemas digitais eletrônicos, a informação binária é representada por tensões (ou correntes) que estão presentes nas entradas e saídas dos vários circuitos. Tipicamente, os 0 e 1 binários são representados por dois níveis de tensão. Por exemplo: zero volts (0 V) poderia representar o binário 0, e +5 V poderia representar o binário 1. Na verdade, devido a variações nos circuitos, os binários 0 e 1 são representados por intervalos de tensão. Isto está mostrado na Fig. 1-7(a), onde qualquer tensão entre 0 e 0,8 V representa o binário 0 e qualquer tensão entre 2 e 5 V representa o binário 1. Todas as entradas e saídas normalmente estarão em um destes intervalos, exceto durante transições de um nível para o outro.

Agora, podemos notar uma significativa diferença entre sistemas analógicos e digitais. Nos sistemas digitais, o valor exato da tensão *não* é importante. Assim, para as tensões mostradas na Fig. 1-7(a), uma tensão de 3,6 V significa o mesmo que uma tensão de 4,3 V. Em sistemas analógicos, o valor exato da tensão *é* importante. Por exemplo, se uma tensão é proporcional à temperatura medida por um transdutor, 3,6 V representaria uma temperatura diferente de 4,3 V. Em outras palavras, o valor da tensão possui informação significativa. Esta característica significa que o projeto de circuitos analógicos precisos é geralmente mais difícil do que o projeto de circuitos digitais, em razão do modo pelo qual os valores exatos de tensões são afetados por variações em valores de componentes, pela temperatura e pelo ruído (flutuações randômicas de tensão).

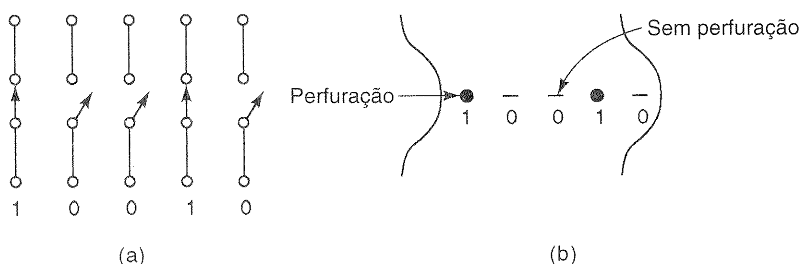


Fig. 1-6 Representação de números binários utilizando (a) chaves e (b) perfurações em uma fita de papel.

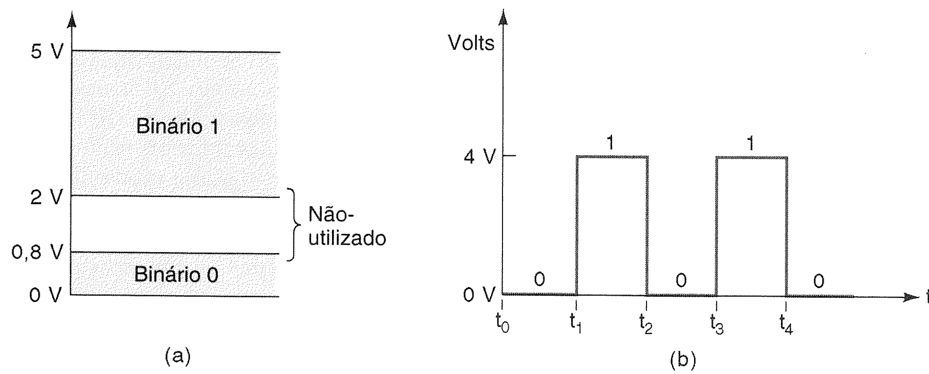


Fig. 1-7 (a) Indicação de intervalos de tensão típicos para binários 0 e 1; (b) típico diagrama de tempo de um sinal digital.

Sinais Digitais e Diagramas de Tempo

A Fig. 1-7(b) mostra um sinal digital típico e como este varia no tempo. Isto é na verdade um gráfico de tensão *versus* tempo *(t)* e é chamado de **diagrama de tempo.** A escala horizontal do tempo é graduada em intervalos regulares, começando em t_0 e depois em t_1, t_2 e assim por diante. Para o exemplo de diagrama de tempo mostrado aqui, o sinal começa em 0 V (0 binário) em t_0 e aí permanece até t_1. Em t_1, o sinal faz uma rápida transição (um salto) até atingir 4 V (1 binário). Em t_2, ele volta a 0 V. Transições semelhantes ocorrem em t_3 e t_4.

As transições no diagrama de tempo são desenhadas como linhas verticais, de modo que elas parecem ser instantâneas quando na realidade não o são. Entretanto, em muitas situações a duração das transições é tão pequena quando comparada com o intervalo de tempo decorrido entre transições que podemos representar essas últimas como linhas verticais no diagrama. Encontraremos mais tarde situações em que será necessário mostrar as transições de modo mais exato, em uma escala de tempo expandida.

Diagramas de tempo são bastante utilizados para mostrar como os sinais digitais variam com o tempo, e especialmente para mostrar a relação entre dois ou mais sinais digitais no mesmo circuito ou sistema. Utilizando um *osciloscópio* ou um *analisador lógico* para observar os sinais digitais, podemos compará-los com o diagrama de tempo esperado. Este procedimento é uma parte muito importante dos métodos de teste e reparo usados em sistemas digitais.

1-5 CIRCUITOS DIGITAIS/CIRCUITOS LÓGICOS

Circuitos digitais são projetados para produzir tensões de saída que estejam dentro dos intervalos determinados para os binários 0 e 1, como aqueles definidos na Fig. 1-7. Da mesma maneira, circuitos digitais são projetados para responder, de modo previsível, a tensões de entrada que estejam dentro dos intervalos definidos para 0 e 1. Isto significa que um circuito digital responderá do mesmo modo a todas as tensões de entrada que estiverem dentro do intervalo permitido para o 0; de maneira semelhante, ele não distinguirá entre tensões de entrada que estejam dentro do intervalo permitido para 1.

Para ilustrar este fato, a Fig. 1-8 representa um típico circuito digital, com entrada v_i e saída v_o. Mostramos a saída

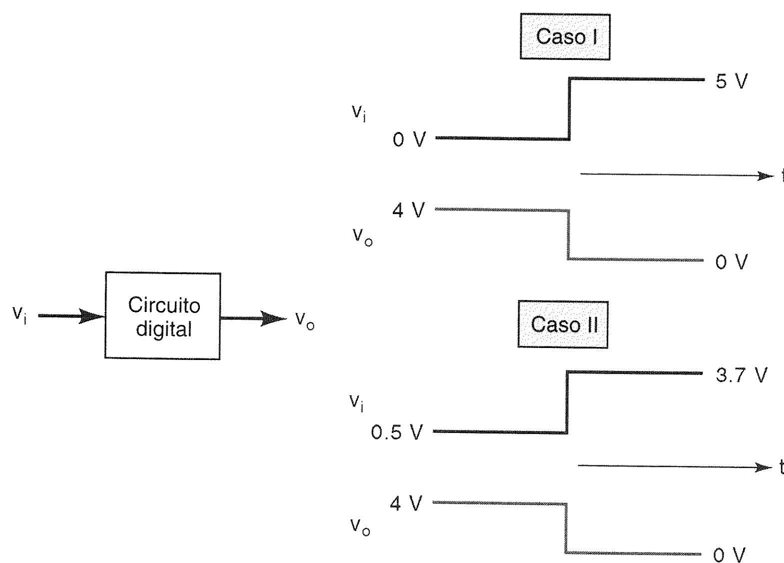


Fig. 1-8 Um circuito digital responde a um nível binário (0 ou 1), e não ao valor exato da tensão de entrada.

correspondente a dois sinais de entrada diferentes. Observe que v_o é o mesmo em ambos os casos, porque embora as duas formas de onda de entrada possuam diferentes valores de tensão, elas possuem o mesmo valor binário.

Circuitos Lógicos

A maneira pela qual um circuito digital responde a uma entrada é chamada de *lógica* do circuito. Cada tipo de circuito digital obedece a um determinado conjunto de regras lógicas. Por esta razão, circuitos digitais também são chamados **circuitos lógicos.** Usaremos ambos os termos de modo intercambiável ao longo do texto. No Cap. 3, veremos mais claramente o que se quer dizer por "lógica" de um circuito.

Estudaremos todos os tipos de circuitos lógicos que são atualmente utilizados em circuitos digitais. Inicialmente, nossa atenção estará concentrada apenas nas operações lógicas que estes circuitos realizam, isto é, estaremos interessados apenas na relação entre a entrada e a saída do circuito. Adiaremos qualquer discussão de como os circuitos lógicos operam internamente até termos desenvolvido uma boa compreensão das suas operações lógicas.

Circuitos Integrados Digitais

Quase todos os circuitos digitais existentes nos sistemas digitais modernos são circuitos integrados (CIs). A grande variedade de CIs lógicos disponível tornou possível a construção de sistemas digitais complexos menores e mais confiáveis do que aqueles construídos com circuitos lógicos discretos.

Existem diversas tecnologias de fabricação utilizadas para produzir CIs digitais, e as mais comuns são: TTL, CMOS, NMOS e ECL. Cada uma difere da outra no tipo de circuitos que são utilizados para obter a operação lógica desejada. Por exemplo, a tecnologia TTL (*Transistor-Transistor Logic*) utiliza o transistor bipolar como elemento principal, enquanto a CMOS (*Complementary Metal Oxide Semiconductor*) usa o MOSFET do tipo enriquecimento como elemento principal. Aprenderemos sobre as várias tecnologias de CIs, suas características, suas vantagens e desvantagens após termos dominado os tipos básicos de circuitos lógicos.

Questão de Revisão

1. *Verdadeiro ou falso*: O valor exato da tensão de entrada é crítico para um circuito digital.
2. Um circuito digital pode produzir a mesma tensão de saída para diferentes valores de tensão de entrada?
3. Um circuito digital também é chamado de circuito ———————.
4. Um gráfico que mostra como um ou mais sinais digitais variam em função do tempo é chamado ——————————.

1-6 TRANSMISSÃO PARALELA E SERIAL

Uma das operações mais comuns que ocorrem em sistemas digitais é a transmissão da informação de um lugar para outro. A informação pode ser transmitida através de uma distância tão pequena quanto alguns centímetros, numa mesma placa de circuito, ou tão grande quanto uma distância de muitos quilômetros quando um operador, num terminal de computador, está se comunicando com um computador em outra cidade. A informação que é transmitida está na forma binária e é geralmente representada por tensões nas saídas de um circuito emissor, que estão conectadas nas entradas de um circuito receptor. A Fig. 1-9 ilustra os dois métodos básicos para transmissão da informação digital: o **paralelo** e o **serial.**

A Fig. 1-9(a) mostra como o número binário 10110 é transmitido do circuito *A* para o circuito *B*, usando transmissão paralela. Cada bit do número binário é representado por uma das saídas do circuito *A*, onde a saída A_4 é o MSB e A_0 é o LSB. Cada uma das saídas do circuito *A* está conectada na entrada correspondente do circuito *B*, portanto todos os cinco bits de informação são transmitidos simultaneamente (em paralelo).

Na Fig. 1-9(b) existe apenas uma conexão do circuito *A* para o circuito *B*, quando a transmissão serial é usada. Neste caso, a saída do circuito *A* produzirá um sinal digital cujo nível de tensão mudará em intervalos regulares de acordo com o número binário transmitido. Desse modo, a informação está sendo transmitida na base de um bit por vez (serialmente), através de uma linha de sinal. O diagrama de tempo na Fig. 1-9(b) mostra como o nível do sinal varia com o tempo. Durante o primeiro intervalo de tempo, T_0, o sinal está no nível 0; durante o intervalo T_1, o sinal está no nível 1, e assim por diante.

O principal compromisso entre as representações paralela e serial está relacionado com a velocidade e a simplicidade do circuito. A transmissão de dados binários, de uma parte de um sistema digital para outra, pode ser feita mais rapidamente usando a representação paralela, porque todos os bits são transmitidos simultaneamente, enquanto a representação serial transmite um bit por vez. Por outro lado, a paralela requer mais linhas de sinal conectadas entre o emissor e o receptor dos dados binários do que a serial. Em outras palavras, a paralela é mais rápida, e a serial requer menos linhas de sinal. Esta comparação entre os métodos paralelo e serial, para representar a informação binária, será encontrada muitas vezes em discussões ao longo do texto.

Questão de Revisão

1. Descreva as vantagens relativas da transmissão paralela e serial de dados binários.

1-7 MEMÓRIA

Quando um sinal de entrada é aplicado na maioria dos dispositivos ou circuitos, a saída de algum modo muda em resposta à entrada, e quando o sinal de entrada é removido a saída retorna ao seu estado original. Estes circuitos não exibem a propriedade de *memória*, já que suas saídas voltam ao normal. Nos circuitos digitais, certos tipos de dispositivos e circuitos têm memória. Quando uma entrada é aplicada em tal circuito, a saída mudará seu estado, mas per-

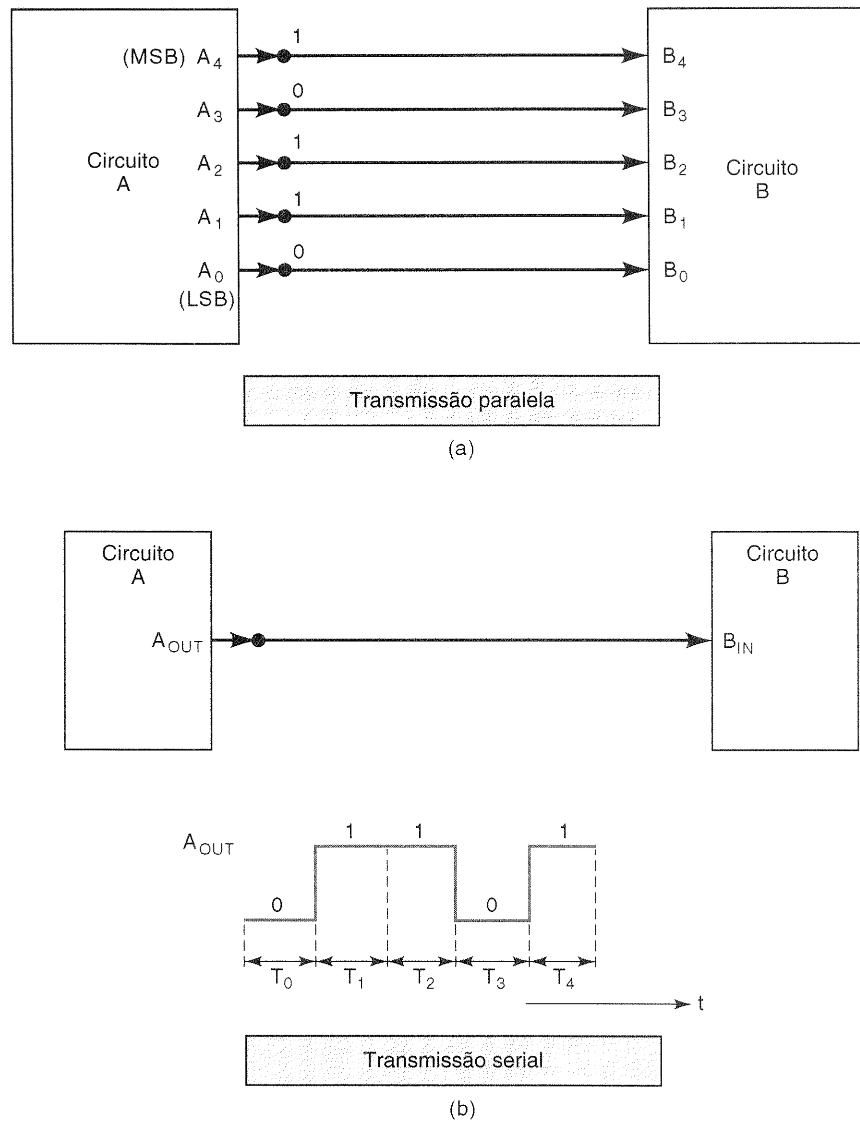


Fig. 1-9 (a) A transmissão paralela utiliza uma linha de conexão por bit, e todos eles são transmitidos simultaneamente; (b) a transmissão serial utiliza apenas uma linha de conexão, na qual cada bit é transmitido serialmente (um bit de cada vez).

manecerá neste novo estado mesmo após a entrada ter sido removida. Esta propriedade de reter sua resposta a uma entrada momentânea é chamada **memória.** A Fig. 1-10 ilustra as operações sem memória e com memória.

Dispositivos e circuitos de memória têm um importante papel em sistemas digitais, porque eles fornecem meios para armazenar números binários tanto temporária quanto permanentemente, com a capacidade de alterar a informação armazenada a qualquer momento. Como veremos, os vários elementos de memória incluem os tipos magnéticos e aqueles que utilizam circuitos eletrônicos de retenção (chamados *latches* e *flip-flops*).

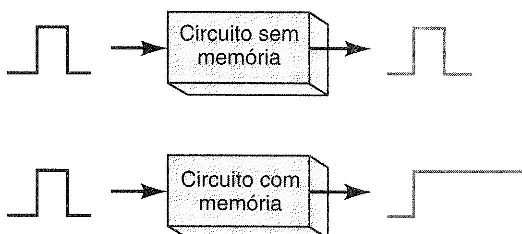


Fig. 1-10 Comparação de operações com e sem memória.

1-8 COMPUTADORES DIGITAIS

As técnicas digitais são aplicadas em inúmeras áreas da tecnologia, mas a área dos **computadores digitais** é de longe a mais ampla e notável. Embora os computadores digitais afetem parte da vida de todos, muitos de nós não sabemos exatamente o que um computador faz. Em termos simples, *um computador é um sistema de hardware que realiza operações aritméticas, manipula dados (usualmente na forma binária) e toma decisões.*

Na maioria das vezes, os seres humanos podem fazer o que os computadores fazem, mas os computadores podem fazê-lo

com uma velocidade e exatidão muito maiores. Isto ocorre apesar de os computadores realizarem todos os cálculos e operações em passos distintos e um de cada vez. Por exemplo: um ser humano pode pegar uma lista de 10 números e calcular a soma geral em uma operação, listando os números um sobre o outro e somando coluna a coluna. Um computador, por outro lado, pode somar os números apenas dois de cada vez; portanto, para somar a mesma lista de números ele executará, na verdade, nove passos de soma. Naturalmente, o fato de o computador necessitar de apenas um microssegundo ou menos para cada passo disfarça esta aparente ineficiência.

Um computador é mais rápido e mais exato do que pessoas, mas, diferentemente da maioria delas, deve ser dado a ele um conjunto completo de instruções que descreva *exatamente* o que fazer a cada passo de sua operação. Este conjunto de instruções, chamado um **programa,** é preparado por uma ou mais pessoas para cada tarefa que o computador deve fazer. Programas são colocados na unidade de memória do computador, codificados em forma binária, tendo cada instrução um código único. O computador toma estes códigos de instruções da memória *um por vez* e realiza a operação associada ao código. Diremos muito mais sobre isto mais adiante.

Partes Principais de um Computador

Existem vários tipos de sistemas computacionais, mas cada um pode ser subdividido nas mesmas unidades funcionais. Cada unidade realiza funções específicas, e todas as unidades funcionam em conjunto para executar as instruções do programa. A Fig. 1-11 mostra as cinco principais partes funcionais de um computador digital e suas interações. As linhas sólidas com setas representam o fluxo de dados e informações. As linhas tracejadas com setas representam o fluxo de sinais de controle e temporização.

As principais funções de cada unidade são:

1. **Unidade de entrada.** Através desta unidade, um conjunto completo de instruções e dados é fornecido para o sistema computacional e para a unidade de memória, para ser armazenado enquanto for preciso. A informação é geralmente apresentada na unidade de entrada por um teclado ou por um disco.
2. **Unidade de memória.** A memória armazena as instruções e os dados recebidos da unidade de entrada. Ela armazena os resultados das operações aritméticas recebidos da unidade aritmética. Ela também fornece informação para a unidade de saída.
3. **Unidade de controle.** Esta unidade toma as instruções da unidade de memória, uma de cada vez, e as interpreta. Ela então gera os sinais apropriados a todas as outras unidades para causar a execução da instrução específica.
4. **Unidade lógica/aritmética.** Todos os cálculos aritméticos e as decisões lógicas são efetuados nesta unidade, cujos resultados podem então ser enviados para a unidade de memória para serem armazenados.
5. **Unidade de saída.** Esta unidade recebe dados da unidade de memória e imprime, mostra, ou então apresenta a informação para o operador (ou processo, no caso de um computador para controle de processos).

Unidade Central de Processamento (CPU — *Central Processing Unit*)

Conforme mostra o diagrama na Fig. 1-11, as unidades de controle e lógica/aritmética são freqüentemente consideradas uma unidade chamada **unidade central de processamento (CPU).** A CPU contém todos os circuitos para buscar e interpretar instruções e para controlar e realizar as várias operações requeridas pelas instruções.

TIPOS DE COMPUTADORES Todos os computadores são formados com as unidades básicas descritas anteriormente, mas podem diferir quanto ao tamanho físico, velocidade de operação, capacidade de memória e poder computacional, dentre outras características. Computadores são freqüentemente classificados de acordo com o tamanho físico, o que geralmente, mas nem sempre, é uma indicação de suas capacidades relativas. As três classificações básicas, do menor para o maior, são: *microcomputador, minicomputador* (*estação de trabalho*) e *mainframe.* Como os microcomputadores têm se tornado mais e mais poderosos, a distinção entre microcomputadores e minicomputadores tornou-se bastante nebulosa, e começou-se a distinguir apenas computadores pequenos — aqueles que cabem num escritório ou numa mesa ou no colo — e computadores grandes — aqueles que são grandes de-

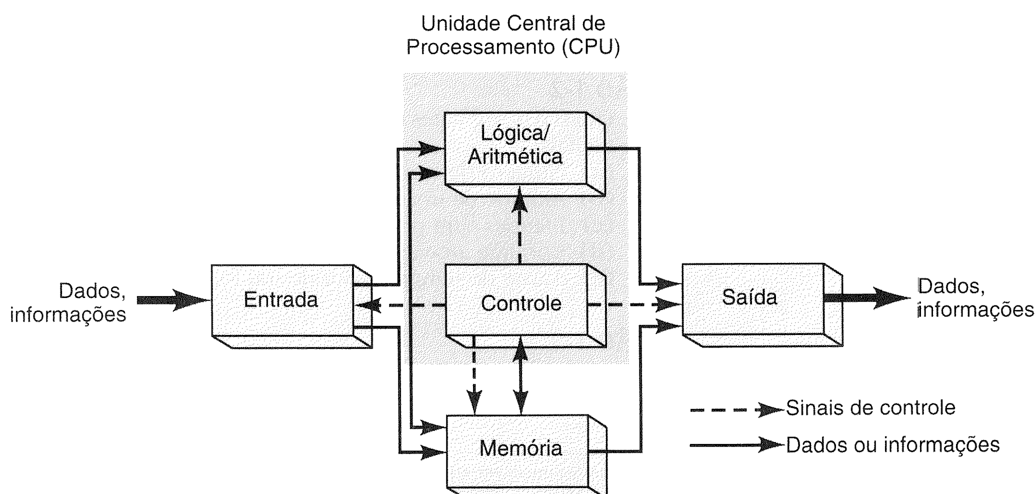


Fig. 1-11 Diagrama funcional de um computador digital.

mais para estes locais. Neste livro, trataremos principalmente dos microcomputadores.

Um **microcomputador** é o menor tipo de computador. Geralmente consiste em várias pastilhas de CIs, incluindo a do **microprocessador,** pastilhas de memória e pastilhas para interface de entrada e saída, além de dispositivos de entrada e saída, tais como teclado, monitor de vídeo, impressora e unidades de disco. Os microcomputadores foram desenvolvidos como resultado dos tremendos avanços na tecnologia de fabricação de CIs, os quais tornaram possível integrar mais e mais circuitos digitais numa pequena pastilha. Por exemplo: a pastilha do microprocessador contém — no mínimo — todos os circuitos que implementam a porção CPU do computador, isto é, a unidade de controle e a unidade lógica/aritmética. O microprocessador, em outras palavras, é uma "CPU em uma pastilha".

A maioria das pessoas está familiarizada com os microcomputadores de propósito geral, como o IBM PC e seus compatíveis, e o Apple Macintosh, que são usados em mais da metade dos lares e em quase todos os ambientes de negócios. Estes microcomputadores podem realizar uma grande variedade de tarefas, numa larga faixa de aplicações, dependendo do software (programas) que eles executam. Existe um tipo mais especializado de microcomputador, denominado **microcontrolador,** que não é um computador de propósito geral. Ao contrário, ele é projetado para ser usado como um *controlador embutido* ou *dedicado,* que ajuda a monitorar e controlar a operação de uma máquina, uma parte de um equipamento ou um processo. Microcontroladores são microcomputadores porque usam uma pastilha de microprocessador como CPU, mas são bem menores do que microcomputadores de propósito geral, porque os dispositivos de entrada e saída que eles usam normalmente são muito menores. De fato, a entrada e a saída — assim como a memória — estão usualmente na mesma pastilha do microprocessador. Estes microcontroladores em uma única pastilha são empregados numa ampla variedade de aplicações de controle, tais como: controle de aparelhos, de máquinas que operam sobre metal, de videocassetes, de máquinas de atendimento automático, de fotocopiadoras, de sistemas de ignição de automóveis, de sistema contra travamento de freios, de instrumentação médica e muito mais.

Questões de Revisão

1. Explique como um circuito digital que tem memória difere de um que não tem.
2. Relacione as cinco principais unidades funcionais de um computador.
3. Quais as duas unidades que compõem a CPU?
4. Uma pastilha de CI que contém uma CPU é chamada um ________.

RESUMO

1. Os dois modos básicos de representar o valor numérico de quantidades físicas são o analógico (contínuo) e o digital (discreto).
2. A maioria das quantidades do mundo real é analógica, mas técnicas digitais são geralmente superiores a técnicas analógicas, e a maioria dos avanços previstos deverá ocorrer no domínio digital.
3. O sistema de numeração binário (0 e 1) é o sistema básico usado em tecnologia digital.
4. Circuitos digitais ou lógicos operam com tensões que ficam restritas a faixas que representam ou 0 binário ou 1 binário.
5. Os dois modos básicos de transferir uma informação digital são o paralelo — todos os bits simultaneamente — e o serial — um bit de cada vez.
6. As partes principais de todos os computadores são as unidades de entrada, controle, memória, lógica/aritmética e de saída.
7. A combinação da unidade lógica/aritmética com a de controle forma a CPU (unidade central de processamento).
8. Um microcomputador usualmente tem uma CPU que está em uma única pastilha denominada *microprocessador.*
9. Um microcontrolador é um microcomputador especialmente projetado para aplicações de controle dedicado (não de propósito geral).

TERMOS IMPORTANTES*

representação analógica
representação digital
sistema digital
sistema analógico
conversor analógico-digital (conversor A/D)
conversor digital-analógico (conversor D/A)
sistema decimal
sistema binário
bit
diagrama de tempo
circuitos digitais/lógicos
paralelo
serial
memória
computador digital
programa
unidade de entrada
unidade de memória
unidade de controle
unidade lógica/aritmética
unidade de saída
unidade central de processamento (CPU)
microcomputador
microprocessador
microcontrolador

PROBLEMAS

SEÇÃO 1-2

1-1. Quais das seguintes quantidades são analógicas e quais são digitais?
- **(a)** Número de átomos numa amostra de material
- **(b)** Altitude de um avião
- **(c)** Pressão num pneu de bicicleta
- **(d)** Corrente através de um alto-falante
- **(e)** Ajuste do temporizador de um forno de microondas

SEÇÃO 1-3

1-2. Converta os números binários a seguir em seus equivalentes valores decimais.

*Estes termos podem ser encontrados em negrito no capítulo e estão definidos no Glossário ao final do livro.

(a) 11001_2
(b) $1001{,}1001_2$
(c) $10011011001{,}10110_2$

1-3. Utilizando seis bits, mostre a seqüência de contagem binária de 000000 até 111111.
1-4. Qual é o número máximo de uma contagem utilizando 10 bits?
1-5. Quantos bits são necessários para contar até um máximo de 511?

SEÇÃO 1-4

1-6. Faça o diagrama de tempo de um sinal digital que continuamente alterna entre 0,2 V (0 binário) durante 2 ms e 4,4 V (1 binário) por 4 ms.

SEÇÃO 1-6

1-7. Suponha que um valor inteiro decimal de 0 a 15 deve ser transmitido.
(a) Quantas linhas serão necessárias se a representação paralela for usada?
(b) Quantas serão necessárias se a representação serial for usada?

SEÇÕES 1-7 E 1-8

1-8. Como um microprocessador difere de um microcomputador?
1-9. Como um microcontrolador difere de um microcomputador?

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 1-1

1. Quantidades analógicas podem assumir *qualquer* valor dentro de uma faixa contínua; quantidades digitais podem assumir apenas valores *discretos*.

SEÇÃO 1-2

1. Mais fácil de projetar; mais fácil de armazenar informação; maior exatidão e precisão; mais fácil de programar; menos afetadas por ruído; maior grau de integração **2.** As quantidades físicas do mundo real são analógicas.

SEÇÃO 1-3

1. 107_{10} **2.** 11000_2 **3.** 4095_{10}

SEÇÃO 1-5

1. Falso **2.** Sim, desde que as duas tensões de entrada estejam dentro da mesma faixa do nível lógico **3.** Lógico **4.** Diagrama de tempo

SEÇÃO 1-6

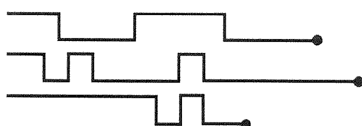
1. A paralela é mais rápida; a serial requer apenas uma linha de sinal.

SEÇÃO 1-8

1. O que tem memória terá sua saída alterada e *permanecerá* alterada em resposta a uma mudança momentânea no sinal de entrada. **2.** Entrada, saída, memória, lógica/aritmética, controle **3.** Controle e lógica/aritmética **4.** Microprocessador

CAPÍTULO 2

Sistemas de Numeração e Códigos



SUMÁRIO

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Usar dois métodos diferentes para realizar conversões de decimal para binário.
- Citar as diversas vantagens dos sistemas de numeração octal e hexadecimal.
- Converter do sistema de numeração hexadecimal ou octal tanto para o decimal quanto para o binário.
- Representar números decimais usando o código BCD.
- Compreender a diferença entre o código BCD e o código binário puro.
- Entender a necessidade dos códigos alfanuméricos, especialmente do código ASCII.
- Descrever o método da paridade para detecção de erros.
- Determinar a paridade (par ou ímpar) de dados digitais.

■ INTRODUÇÃO

O sistema de numeração binário é o mais importante em sistemas digitais, mas muitos outros também o são. O sistema decimal é importante porque é universalmente usado para representar quantidades externas a um sistema digital. Isto significa que existem situações nas quais valores decimais devem ser convertidos para valores binários antes de entrarem num sistema digital. Por exemplo, quando você digita um número decimal na calculadora (ou computador), os circuitos internos do dispositivo convertem o número decimal num valor binário.

Do mesmo modo, existem situações em que os valores binários nas saídas de um sistema digital devem ser convertidos em valores decimais para apresentação ao mundo exterior. Por exemplo, sua calculadora (ou computador) utiliza números binários para calcular as respostas de um problema e então converte-as para decimal antes de apresentá-las.

Além dos sistemas binário e decimal, dois outros sistemas de numeração encontram aplicações em diversas áreas de sistemas digitais. Os sistemas de numeração *octal* (base 8) e *hexadecimal* (base 16) são usados para o mesmo propósito — fornecer um meio eficiente de representar números binários grandes. Como veremos, os dois sistemas têm a vantagem de ser facilmente convertidos para o sistema binário e vice-versa.

Num sistema digital, três ou quatro desses sistemas de numeração podem estar em uso ao mesmo tempo, portanto o entendimento da operação do sistema requer a habilidade de converter de um sistema numérico para outro. Este capítulo apresenta como realizar estas conversões. Embora alguns deles não sejam de uso imediato no nosso estudo de sistemas digitais, você precisará deles quando começar a estudar microprocessadores.

Este capítulo também introduz alguns dos *códigos binários* que são usados para representar vários tipos de informação. Estes códigos binários utilizam 1s e 0s, mas de um modo que difere do sistema de numeração binário.

2-1 CONVERSÕES BINÁRIO-DECIMAL

Conforme explicado no Cap. 1, o sistema de numeração binário é um sistema posicional em que cada dígito binário (bit) tem um certo peso de acordo com sua posição relativa ao LSB. Qualquer número binário pode ser convertido para o seu equivalente decimal simplesmente somando-se os pesos das várias posições que contiverem 1 no número binário. Para ilustrar:

$$\begin{array}{l} 1 \quad 1 \quad 0 \quad 1 \quad 1_2 \qquad (\textit{binário}) \\ 2^4 + 2^3 + 0 + 2^1 + 2^0 = 16 + 8 + 2 + 1 \\ \qquad = 27_{10} \quad (\textit{decimal}) \end{array}$$

Vejamos outro exemplo com um maior número de bits:

$$\begin{array}{l} 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1_2 = \\ 2^7 + 0 + 2^5 + 2^4 + 0 + 2^2 + 0 + 2^0 = 181_{10} \end{array}$$

Note que o procedimento é determinar os pesos (isto é, as potências de 2) para cada bit que contém 1, e então somá-los. Note também que o MSB tem um peso de 2^7, embora seja o oitavo bit; isto acontece porque o LSB é o primeiro bit e tem um peso de 2^0.

Questões de Revisão

1. Converta 100011011011_2 para seu equivalente decimal.
2. Qual é o peso do MSB de um número de 16 bits?

2-2 CONVERSÕES DECIMAL-BINÁRIO

Há dois modos de converter um número decimal *inteiro* para a representação equivalente no sistema binário. O primeiro método é o inverso do processo descrito na Seção 2-1. O número decimal é simplesmente representado como uma soma de potências de 2, e então 1s e 0s são escritos nas posições de bit apropriadas. Para ilustrar:

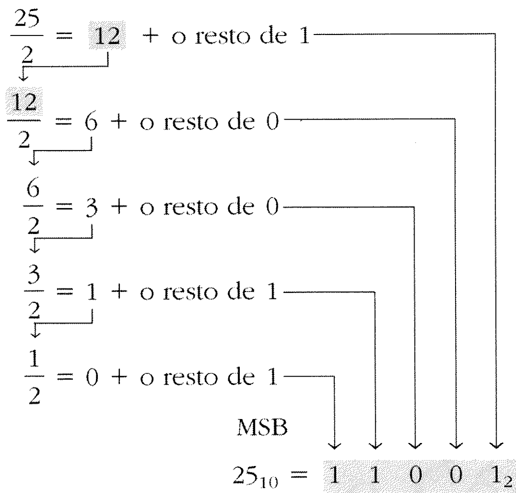
$$\begin{array}{l} 45_{10} = 32 + 8 + 4 + 1 = 2^5 + 0 + 2^3 + 2^2 + 0 + 2^0 \\ \qquad = 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1_2 \end{array}$$

Note que um 0 é colocado nas posições de 2^1 e 2^4, já que todas as posições devem ser levadas em conta. Um outro exemplo é o seguinte:

$$\begin{array}{l} 76_{10} = 64 + 8 + 4 = 2^6 + 0 + 0 + 2^3 + 2^2 + 0 + 0 \\ \qquad = 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0_2 \end{array}$$

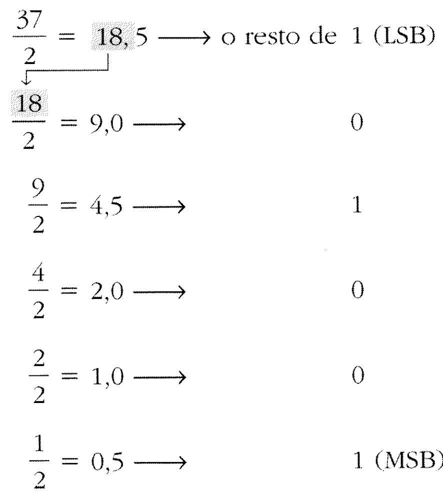
Divisões Sucessivas

Um outro método para converter números decimais inteiros utiliza sucessivas divisões por 2. A conversão, ilustrada adiante para 25_{10}, requer repetidas divisões do número decimal por 2 e a escrita do resto de cada divisão até que o quociente 0 seja obtido. Note que o resultado binário é obtido escrevendo-se o primeiro resto como o LSB e o último resto como o MSB.



Este processo, mostrado no fluxograma da Fig. 2-1, também pode ser usado para converter de decimal para qualquer outro sistema de numeração.

Se uma calculadora for usada para realizar as divisões por 2, os restos podem ser determinados notando se o quociente tem ou não uma parte fracionária. Por exemplo, a calculadora deveria produzir 25/2 = 12,5. O ",5" indica que existe um resto 1. A calculadora deveria também produzir 12/2 = 6,0, que indica um resto 0. No exemplo seguinte é mostrado o que ocorreria usando-se uma calculadora.



Logo, $37_{10} = 100101_2$.

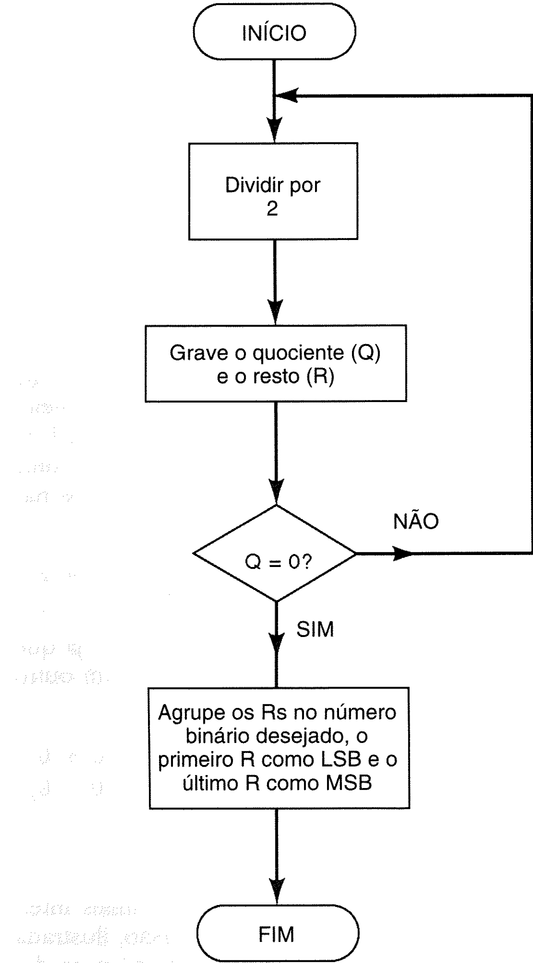


Fig. 2-1 Fluxograma do método das divisões sucessivas para a conversão decimal-binário de inteiros. O mesmo processo pode ser usado para converter um inteiro decimal para qualquer outro sistema de numeração.

Faixa de Contagem

Lembre-se de que usando *N* bits podemos contar 2^N valores decimais diferentes variando de 0 até $2^N - 1$. Por exemplo, para $N = 4$, podemos contar de 0000_2 até 1111_2, ou seja, de 0_{10} até 15_{10}, totalizando 16 números diferentes. O maior valor decimal é $2^4 - 1 = 15$, e existem 2^4 números diferentes.

Portanto, de um modo geral, podemos afirmar:

Usando N bits, podemos representar valores decimais variando de 0 até 2^N 2 1, num total de 2^N valores.

EXEMPLO 2-1

(a) Qual é a faixa de valores decimais que pode ser representada com oito bits?

(b) Quantos bits são necessários para representar valores decimais variando de 0 até 12.500?

Solução

(a) Aqui temos $N = 8$. Logo, podemos representar números decimais desde **0** até $2^8 - 1 = 255$. Podemos verificar isto constatando que 11111111_2 equivale a 255_{10}.

(b) Com 13 bits, podemos contar de 0 até $2^{13} - 1 = 8.191$. Com 14 bits, podemos contar de 0 até $2^{14} - 1 = 16.383$. Claramente, 13 bits não são suficientes, e com 14 bits obtemos além de 12.500. Portanto, o número de bits necessário é 14.

Questões de Revisão

1. Converta 83_{10} para binário utilizando ambos os métodos.
2. Converta 729_{10} para binário utilizando ambos os mé-

todos. Verifique sua resposta convertendo de volta para decimal.
3. Quantos bits são necessários para contar até 1 milhão em decimal?

2-3 SISTEMA DE NUMERAÇÃO OCTAL

O sistema de numeração octal é muito importante no trabalho com computadores digitais. O **sistema de numeração octal** tem base *oito*, significando que tem oito dígitos possíveis: 0, 1, 2, 3, 4, 5, 6 e 7. Assim, cada dígito de um número octal pode ter valores de 0 a 7. As posições dos dígitos num número octal têm pesos, como segue:

8^4	8^3	8^2	8^1	8^0	8^{-1}	8^{-2}	8^{-3}	8^{-4}	8^{-5}

ponto octal (,)

Conversão Octal-Decimal

Um número octal pode ser facilmente convertido para seu equivalente decimal multiplicando-se cada dígito octal pelo seu peso posicional. Por exemplo:

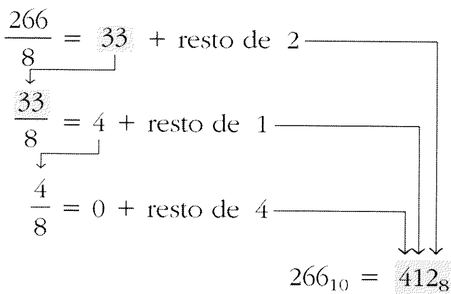
$$\begin{aligned} 372_8 &= 3 \times (8^2) + 7 \times (8^1) + 2 \times (8^0) \\ &= 3 \times 64 + 7 \times 8 + 2 \times 1 \\ &= 250_{10} \end{aligned}$$

Eis outro exemplo:

$$\begin{aligned} 24{,}6_8 &= 2 \times (8^1) + 4 \times (8^0) + 6 \times (8^{-1}) \\ &= 20{,}75_{10} \end{aligned}$$

Conversão Decimal-Octal

Um inteiro decimal pode ser convertido para octal utilizando o mesmo método das divisões sucessivas que foi usado na conversão decimal-binário (Fig. 2-1), mas com o fator de divisão 8 em vez de 2. Um exemplo é mostrado a seguir.



Note que o primeiro resto se torna o dígito menos significativo (LSD) do número octal, e o último resto se torna o dígito mais significativo (MSD).

Se uma calculadora é usada para realizar as divisões no processo anterior, o resultado incluirá uma fração decimal em vez de um resto. O resto pode ser obtido multiplicando-se a fração decimal por 8. Por exemplo, 266/8 produz 33,25. O resto é $0{,}25 \times 8 = 2$. Analogamente, 33/8 é 4,125, e o resto se torna $0{,}125 \times 8 = 1$.

Conversão Octal-Binário

A principal vantagem do sistema de numeração octal é a facilidade com que conversões podem ser feitas entre números binários e octais. A conversão de octal para binário é realizada convertendo-se *cada* dígito octal nos três bits binários equivalentes. Os oito dígitos possíveis são convertidos conforme indicado na Tabela 2-1.

TABELA 2-1

Dígito Octal	0	1	2	3	4	5	6	7
Equivalente Binário	000	001	010	011	100	101	110	111

Usando essas conversões, podemos converter qualquer número octal para binário convertendo individualmente cada dígito. Por exemplo, podemos converter 472_8 para binário como segue:

$$\begin{array}{ccc} 4 & 7 & 2 \\ \downarrow & \downarrow & \downarrow \\ 100 & 111 & 010 \end{array}$$

Portanto, o octal 472 é equivalente ao binário 100111010. Como outro exemplo, considere a conversão de 5431_8 para binário:

$$\begin{array}{cccc} 5 & 4 & 3 & 1 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 101 & 100 & 011 & 001 \end{array}$$

Assim, $5431_8 = 101100011001_2$.

Conversão Binário-Octal

Converter binários inteiros para octais inteiros é simplesmente o inverso do processo anterior. Os bits do número binário são reunidos em grupos de *três* bits iniciando-se do LSB. Então cada grupo é convertido para seu equivalente octal (Tabela 2-1). Para ilustrar, considere a conversão de 100111010_2 para octal.

$$\begin{array}{ccc} \underbrace{1\ 0\ 0} & \underbrace{1\ 1\ 1} & \underbrace{0\ 1\ 0} \\ \downarrow & \downarrow & \downarrow \\ 4 & 7 & 2_8 \end{array}$$

Algumas vezes, o número binário não tem grupos completos de três bits. Nesses casos, podemos adicionar um ou dois 0s à esquerda do MSB do número binário para preencher o último grupo. Isto é ilustrado adiante para o número binário 11010110.

$$\begin{array}{ccc} \underbrace{0\ 1\ 1} & \underbrace{0\ 1\ 0} & \underbrace{1\ 1\ 0} \\ \downarrow & \downarrow & \downarrow \\ 3 & 2 & 6_8 \end{array}$$

Note que um 0 foi colocado à esquerda do MSB para produzir grupos completos de três bits.

Contando em Octal

O maior dígito octal é 7, portanto na contagem em octal cada posição de dígito é incrementada de 0 a 7. Uma vez

alcançado o 7, ele retorna para 0 na próxima contagem e causa o incremento da próxima posição de dígito mais alta. Isto é ilustrado nas seguintes seqüências de contagem octal: (1) 65, 66, 67, 70, 71 e (2) 275, 276, 277, 300.

Com *N* posições de dígitos octais, podemos contar de 0 até $8^N - 1$, para um total de 8^N valores diferentes. Por exemplo, com três posições de dígitos octais podemos contar de 000_8 até 777_8, ou seja, de 0_{10} até 511_{10} para um total de $8^3 = 512_{10}$ números octais diferentes.

Utilidade do Sistema Octal

A facilidade com que as conversões podem ser feitas entre octal e binário torna o sistema octal atrativo como um modo "compacto" de expressar números binários grandes. No trabalho com computadores, números binários com até 64 bits não são incomuns. Estes números binários, conforme veremos, nem sempre representam uma quantidade numérica, mas são algum tipo de código que carregam freqüentemente informação não-numérica. Nos computadores, números binários podem representar (1) dados numéricos puros, (2) números correspondentes a posições (endereços) de memória, (3) um código de instrução, (4) um código representando caracteres alfabéticos e outros não-numéricos, ou (5) um grupo de bits representando o estado de dispositivos internos ou externos ao computador.

Quando lidamos com uma grande quantidade de números binários de vários bits, é conveniente e mais eficiente escrevermos os números em octal em vez de binário. Não devemos esquecer, no entanto, que circuitos e sistemas digitais trabalham exclusivamente em binário; usamos octal somente por conveniência para os operadores do sistema.

EXEMPLO 2-2

Converta 177_{10} para seu equivalente binário de oito bits, convertendo primeiramente para octal.

Solução

$$\frac{177}{8} = 22 + \text{resto de } 1$$

$$\frac{22}{8} = 2 + \text{resto de } 6$$

$$\frac{2}{8} = 0 + \text{resto de } 2$$

Assim, $177_{10} = 261_8$. Agora podemos converter este número octal para seu equivalente binário 010110001_2, e finalmente temos

$$177_{10} = 10110001_2$$

Note que descartamos o 0 à esquerda para expressar o resultado com 8 bits.

Este método de conversão decimal-octal-binário freqüentemente é mais rápido do que converter diretamente de decimal para binário, sobretudo para números grandes. De modo semelhante, freqüentemente é mais rápido converter de binário para decimal convertendo primeiro para octal.

Questões de Revisão

1. Converta 614_8 para decimal.
2. Converta 146_{10} para octal, e então de octal para binário.
3. Converta 10011101_2 para octal.
4. Escrever os três próximos números nesta seqüência de contagem octal: 624, 625, 626, ____, ____, ____.
5. Converta 975_{10} para binário, convertendo-o primeiramente para octal.
6. Converta o binário 1010111011 para decimal, convertendo-o primeiramente para octal.
7. Qual é a faixa de valores decimais que pode ser representada por um número octal de quatro dígitos?

2-4 SISTEMA DE NUMERAÇÃO HEXADECIMAL

O **sistema de numeração hexadecimal** usa a base 16. Assim, ele tem 16 símbolos possíveis. Ele usa os dígitos 0 a 9 mais as letras A, B, C, D, E e F como os 16 símbolos. A Tabela 2-2 mostra as relações entre hexadecimal, decimal e binário. Note que cada dígito hexadecimal representa um grupo de quatro dígitos binários. É importante lembrar que os dígitos hexa (abreviatura de "hexadecimal") A até F são equivalentes aos valores decimais 10 até 15.

TABELA 2-2

Hexadecimal	Decimal	Binário
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Conversão Hexadecimal-Decimal

Um número hexa pode ser convertido para seu equivalente decimal usando o fato de que cada posição de dígito hexa tem um peso que é uma potência de 16. O LSD tem um peso de $16^0 = 1$; a próxima posição de dígito mais alta tem um peso de $16^1 = 16$; a próxima tem um peso de $16^2 = 256$; e assim por diante. O processo de conversão é demonstrado nos exemplos a seguir:

$$356_{16} = 3 \times 16^2 + 5 \times 16^1 + 6 \times 16^0$$
$$= 768 + 80 + 6$$
$$= 854_{10}$$

$$2AF_{16} = 2 \times 16^2 + 10 \times 16^1 + 15 \times 16^0$$
$$= 512 + 160 + 15$$
$$= 687_{10}$$

Note que no segundo exemplo o valor 10 substituiu o A e o valor 15 o F na conversão para decimal.

Para praticar, verifique que $1BC2_{16}$ é igual a 7106_{10}.

Conversão Decimal-Hexadecimal

Relembre que fizemos conversões decimal-binário usando sucessivas divisões por 2, e decimal-octal usando sucessivas divisões por 8. Do mesmo modo, conversões decimal-hexadecimal podem ser feitas usando sucessivas divisões por 16 (Fig. 2-1). Os exemplos seguintes ilustram o método.

EXEMPLO 2-3

Converta 423_{10} para hexa.

Solução

$$\frac{423}{16} = 26 + \text{resto de } 7$$

$$\frac{26}{16} = 1 + \text{resto de } 10$$

$$\frac{1}{16} = 0 + \text{resto de } 1$$

$$423_{10} = 1A7_{16}$$

EXEMPLO 2-4

Converta 214_{10} para hexa.

Solução

$$\frac{214}{16} = 13 + \text{resto de } 6$$

$$\frac{13}{16} = 0 + \text{resto de } 13$$

$$214_{10} = D6_{16}$$

Repare novamente que os restos do processo de divisão formam os dígitos do número hexa. Note também que qualquer resto maior do que 9 é representado pelas letras de A até F.

Se uma calculadora está sendo usada para realizar as divisões do processo de conversão, os resultados incluirão uma fração decimal em vez de um resto. O resto pode ser obtido multiplicando-se a fração por 16. Para ilustrar, no Exemplo 2-4 uma calculadora produziria

$$\frac{214}{16} = 13{,}375$$

O resto se torna $(0{,}375) \times 16 = 6$.

Conversão Hexadecimal-Binário

Assim como o sistema de numeração octal, o sistema de numeração hexadecimal é usado principalmente como um método "compacto" para representação de números binários. É relativamente simples converter um número hexa em binário. *Cada* dígito hexa é convertido para seu equivalente de quatro bits (Tabela 2-2). Isto é ilustrado a seguir para $9F2_{16}$.

$$9F2_{16} = \quad 9 \qquad F \qquad 2$$
$$\downarrow \qquad \downarrow \qquad \downarrow$$
$$= 1\ 0\ 0\ 1 \quad 1\ 1\ 1\ 1 \quad 0\ 0\ 1\ 0$$
$$= 100111110010_2$$

Para praticar, verifique que $BA6_{16} = 101110100110_2$.

Conversão Binário-Hexadecimal

A conversão de binário para hexa é apenas o inverso do processo anterior. O número binário é reunido em grupos de *quatro* bits, e cada grupo é convertido para seu equivalente dígito hexa. Zeros são adicionados, se necessário, para completar um grupo de quatro bits (vide sombreado).

$$1\,1\,1\,0\,1\,0\,0\,1\,1\,0_2 = \underbrace{0\,0\,1\,1}_{3}\ \underbrace{1\,0\,1\,0}_{A}\ \underbrace{0\,1\,1\,0}_{6}$$
$$= 3A6_{16}$$

De modo a realizar estas conversões entre hexa e binário, é necessário saber a equivalência entre os números binários de quatro bits (0000 até 1111) e os dígitos hexa. Uma vez dominadas, as conversões podem ser realizadas rapidamente sem necessidade de cálculos. Isto explica por que o hexa (e o octal) são tão úteis na representação de números binários grandes.

Para praticar, verifique que $101011111_2 = 15F_{16}$.

Contando em Hexadecimal

Quando contamos em hexa, cada posição de dígito pode ser incrementada (aumentada de 1) de 0 até F. Uma vez que uma posição de dígito alcance o valor F, ela volta a 0, e a próxima posição de dígito é incrementada. Isto é ilustrado nas seguintes seqüências de contagem hexa:

(a) 38, 39, 3A, 3B, 3C, 3D, 3E, 3F, 40, 41, 42
(b) 6F8, 6F9, 6FA, 6FB, 6FC, 6FD, 6FE, 6FF, 700

Note que quando existe um 9 numa posição de dígito, ele se torna um A quando é incrementado.

Com *N* posições de dígitos hexa podemos contar de 0 a $16^N - 1$ em decimal, para um total de 16^N valores diferentes. Por exemplo, com três dígitos hexa podemos contar de 000_{16} até FFF_{16}, que é de 0_{10} até 4095_{10}, para um total de $4096 = 16^3$ valores diferentes.

EXEMPLO 2-5

Converta o decimal 378 para um binário de 16 bits, primeiramente convertendo-o para hexa.

Solução

$$\frac{378}{16} = 23 + \text{resto de } 10$$

$$\frac{23}{16} = 1 + \text{resto de } 7$$

$$\frac{1}{16} = 0 + \text{resto de } 1$$

Logo, $378_{10} = 17A_{16}$. Este valor hexa pode ser facilmente convertido para o binário 000101111010. Finalmente, podemos expressar 378_{10} como um número binário de 16 bits adicionando-se quatro 0s à esquerda:

$$378_{10} = 0000 \quad 0001 \quad 0111 \quad 1010_2$$

EXEMPLO 2-6

Converta $B2F_{16}$ para octal.

Solução

É mais fácil primeiro converter de hexa para binário, e então para octal.

$B2F_{16}$	=	1011	0010	1111		{converter para binário}
	=	101	100	101	111	{reunir em grupo de três bits}
	=	5	4	5	7_8	{converter para octal}

Resumo das Conversões

Neste ponto, sua cabeça provavelmente está rodando enquanto você tenta guardar todos estes sistemas — binário, decimal, octal, hexa — e todas as diferentes conversões de um para o outro. Você pode não acreditar, mas à medida que você usar mais e mais estes vários sistemas, você acabará conhecendo-os muito bem. Por enquanto, o seguinte resumo deve ajudá-lo a fazer as diferentes conversões:

1. Quando converter de binário [ou octal ou hexa] para decimal, use o método da soma ponderada para cada posição de dígito.
2. Quando converter de decimal para binário [ou octal ou hexa], use o método das divisões sucessivas por 2 [ou 8 ou 16], agrupando os restos (Fig. 2-1).
3. Quando converter de binário para octal [ou hexa], reúna os bits em grupos de três [ou quatro] e converta cada grupo no dígito octal [ou hexa] correto.
4. Quando converter de octal [ou hexa] para binário, converta cada dígito para o seu equivalente de três [ou quatro] bits.
5. Quando converter de octal para hexa [ou vice-versa], primeiramente converta para binário; então converta o binário para o sistema de numeração desejado.

Questões de Revisão

1. Converta $24CE_{16}$ para decimal.
2. Converta 3117_{10} para hexa, e depois para binário.
3. Converta 1001011110110101_2 para hexa.
4. Escreva os próximos quatro números nesta seqüência de contagem hexa: E9A, E9B, E9C, E9D, ____, ____, ____, ____.
5. Converta 3527_8 para hexa.
6. Qual é a faixa de valores decimais que pode ser representada por um número hexa de quatro dígitos?

2-5 CÓDIGO BCD

Quando números, letras ou palavras são representados por um grupo especial de símbolos, dizemos que estão codificados, e o grupo de símbolos é chamado de *código*. Provavelmente um dos códigos mais conhecidos é o código Morse, em que uma série de traços e pontos representam as letras do alfabeto.

Já vimos que qualquer número decimal pode ser representado por um número binário equivalente. O grupo de 0s e 1s no número binário pode ser imaginado como um código representando o número decimal. Quando um número decimal é representado por seu número binário equivalente, denomina-se **codificação binária pura.**

Todos os sistemas digitais utilizam alguma forma de números binários para suas operações internas, mas o mundo exterior é decimal por natureza. Isto significa que conversões entre os sistemas decimal e binário são realizadas freqüentemente. Vimos que conversões entre decimal e binário podem se tornar longas e complicadas para números grandes. Por essa razão, um meio de codificar números decimais que combina algumas características tanto do sistema decimal quanto do sistema binário é usado em certas situações.

Código Decimal Codificado em Binário

Se *cada* dígito de um número decimal é representado por seu equivalente binário, o resultado é um código chamado **decimal codificado em binário** (daqui para a frente abreviado como BCD, do inglês Binary-Coded-Decimal). Como um dígito decimal pode assumir o valor 9, quatro bits são necessários para codificar cada dígito (o código binário para 9 é 1001).

Para ilustrar o código BCD, considere um número decimal como 874. Cada *dígito* é substituído pelo seu equivalente binário do seguinte modo:

8	7	4	(decimal)
↓	↓	↓	
1000	0111	0100	(BCD)

Como um outro exemplo, vamos transformar 943 para sua representação no código BCD:

9	4	3	(decimal)
↓	↓	↓	
1001	0100	0011	(BCD)

Mais uma vez, cada dígito decimal é trocado pelo seu binário equivalente puro. Note que *sempre* são usados quatro bits para cada dígito.

O código BCD, portanto, representa cada dígito do número decimal por um número binário de quatro bits. Obviamente apenas os números binários de quatro bits de 0000 até 1001 são usados. O código BCD não utiliza os números 1010, 1011, 1100, 1101, 1110 e 1111. Em outras palavras, somente 10 dos 16 grupos possíveis de quatro bits são usados. Se algum número de quatro bits "proibido" ocorrer numa máquina usando o código BCD, usualmente é uma indicação de que um erro aconteceu.

EXEMPLO 2-7

Converta 0110100000111001 (BCD) para seu equivalente decimal.

Solução

Divida o número BCD em grupos de quatro bits e converta cada um deles para decimal.

$$\underbrace{0110}_{6}\ \underbrace{1000}_{8}\ \underbrace{0011}_{3}\ \underbrace{1001}_{9}$$

EXEMPLO 2-8

Converta o número BCD 011111000001 para seu equivalente decimal.

Solução

$$\underbrace{0111}_{7}\ \underset{\downarrow}{1100}\ \underbrace{0001}_{1}$$

Este grupo de bits é proibido e indica um erro no número BCD.

Comparação entre BCD e Binário

É importante ressaltar que o BCD não é um outro sistema de numeração tal como o binário, o octal, o decimal ou o hexadecimal. Ele é, na verdade, um sistema decimal com cada dígito codificado no seu equivalente binário. Também é importante compreender que um número BCD *não* é o mesmo que um número binário puro. O código binário puro considera o número decimal *completo* e o representa em binário; o código BCD converte *cada dígito* decimal para binário individualmente. Para ilustrar, considere o número 137 e compare seus códigos binário puro e BCD:

$137_{10} = 10001001_2$ (binário)
$137_{10} = 0001\ 0011\ 0111$ (BCD)

O código BCD requer 12 bits, enquanto o código binário puro necessita de apenas 8 bits para representar 137. O BCD necessita de mais dígitos do que o binário puro para representar números decimais com mais de um dígito. Isto é porque o BCD não usa todos os grupos possíveis de quatro bits, conforme ressaltado anteriormente, e por isso é um tanto ineficiente.

A principal vantagem do código BCD é a relativa facilidade de conversão para o decimal e vice-versa. Apenas os códigos de quatro bits para os dígitos decimais de 0 até 9 precisam ser lembrados. Esta facilidade de conversão é especialmente importante sob o ponto de vista do hardware porque num sistema digital são os circuitos lógicos que realizam as conversões de e para decimal.

Questões de Revisão

1. Represente o valor decimal 178 pelo seu equivalente binário puro. Depois codifique o mesmo número usando BCD.
2. Quantos bits são necessários para representar um número decimal de oito dígitos em BCD?
3. Qual é a vantagem de codificar um número decimal em BCD quando comparado com binário puro? Qual é a desvantagem?

2-6 RELACIONANDO AS REPRESENTAÇÕES

A Tabela 2-3 mostra a representação dos números decimais de 0 até 15 nos sistemas de numeração binário, octal, hexadecimal e no código BCD. Examine-a cuidadosamente e esteja certo de compreender como ela foi obtida. Observe especialmente como a representação BCD sempre usa quatro bits para cada dígito decimal.

TABELA 2-3

Decimal	Binário	Octal	Hexadecimal	BCD
0	0	0	0	0000
1	1	1	1	0001
2	10	2	2	0010
3	11	3	3	0011
4	100	4	4	0100
5	101	5	5	0101
6	110	6	6	0110
7	111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101

2-7 O BYTE

A maioria dos microcomputadores manipula e armazena dados binários e informações em grupos de oito bits; assim, um nome especial é dado para uma cadeia (ou seqüência) de oito bits: é o chamado **byte**. Um byte sempre corresponde a oito bits, e pode representar numerosos tipos de dados ou informações. Os exemplos seguintes ilustram isso.

EXEMPLO 2-9

Quantos bytes existem numa cadeia de 32 bits?

Solução

32/8 = 4, logo existem **quatro** bytes numa cadeia de 32 bits.

EXEMPLO 2-10

Qual é o maior valor decimal que pode ser representado em binário usando dois bytes?

Solução

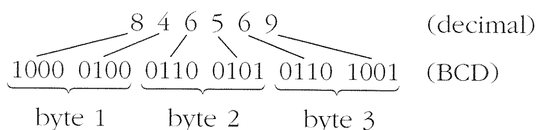
Dois bytes são 16 bits, logo o maior valor binário será equivalente ao decimal $2^{16} - 1 = 65.535$.

EXEMPLO 2-11

Quantos bytes são necessários para representar o valor decimal 846.569 em BCD?

Solução

Cada dígito decimal é convertido para um código BCD de quatro bits. Assim, um número decimal de seis dígitos requer 24 bits. Estes 24 bits equivalem a **três** bytes. Isto é ilustrado a seguir.



Questões de Revisão

1. Quantos bytes são necessários para representar 235_{10} em binário?
2. Qual é o maior valor decimal que pode ser representado em BCD usando-se dois bytes?

2-8 CÓDIGOS ALFANUMÉRICOS

Além de dados numéricos, um computador deve ser capaz de manipular informação não-numérica. Em outras palavras, um computador deve reconhecer códigos que representam letras do alfabeto, sinais de pontuação e outros caracteres especiais, como também os números. Estes códigos são chamados de **códigos alfanuméricos.** Um código alfanumérico completo deve incluir as 26 letras minúsculas, as 26 letras maiúsculas, 10 dígitos numéricos, 7 sinais de pontuação, e entre 20 e 40 outros caracteres, tais como +, /, #, %, *, e assim por diante.* Podemos dizer que um código alfanumérico representa todos os caracteres e funções encontrados num teclado de computador.

*N. do T.: Considerações feitas para a língua inglesa.

Código ASCII

O código alfanumérico mais amplamente usado é o **American Standard Code for Information Interchange (ASCII).** O código ASCII (pronuncia-se "asquii") é um código de sete bits, e portanto tem $2^7 = 128$ codificações possíveis. Isto é mais do que suficiente para representar todos os caracteres de um teclado padrão, como também as funções de controle, tais como as funções de <RETURN> e <LINEFEED>. A Tabela 2-4 mostra uma listagem parcial do código ASCII. Além do código binário para cada caracter, a tabela apresenta os equivalentes octal e hexadecimal.

TABELA 2-4 Listagem parcial do código ASCII.

Caracter	ASCII de sete bits	Octal	Hexa
A	100 0001	101	41
B	100 0010	102	42
C	100 0011	103	43
D	100 0100	104	44
E	100 0101	105	45
F	100 0110	106	46
G	100 0111	107	47
H	100 1000	110	48
I	100 1001	111	49
J	100 1010	112	4A
K	100 1011	113	4B
L	100 1100	114	4C
M	100 1101	115	4D
N	100 1110	116	4E
O	100 1111	117	4F
P	101 0000	120	50
Q	101 0001	121	51
R	101 0010	122	52
S	101 0011	123	53
T	101 0100	124	54
U	101 0101	125	55
V	101 0110	126	56
W	101 0111	127	57
X	101 1000	130	58
Y	101 1001	131	59
Z	101 1010	132	5A
0	011 0000	060	30
1	011 0001	061	31
2	011 0010	062	32
3	011 0011	063	33
4	011 0100	064	34
5	011 0101	065	35
6	011 0110	066	36
7	011 0111	067	37
8	011 1000	070	38
9	011 1001	071	39
espaço	010 0000	040	20
.	010 1110	056	2E
(	010 1000	050	28
+	010 1011	053	2B
$	010 0100	044	24
*	010 1010	052	2A
)	010 1001	051	29
—	010 1101	055	2D
/	010 1111	057	2F
,	010 1100	054	2C
=	011 1101	075	3D
<RETURN>	000 1101	015	0D
<LINEFEED>	000 1010	012	0A

EXEMPLO 2-12

A mensagem a seguir é uma mensagem codificada em código ASCII. Qual é a mensagem?

1001000 1000101 1001100 1010000

Solução

Converta cada código de sete bits para seu hexa equivalente. Os resultados são

48 45 4C 50

Agora localize estes valores hexa na Tabela 2-4 e determine o caracter representado por cada um. Os resultados são

H E L P

O código ASCII é usado para a transferência de informações entre um computador e dispositivos de entrada e saída como terminais de vídeo e impressoras. Um computador também o utiliza internamente para armazenar informações que um operador digita no teclado. O exemplo seguinte ilustra isso.

EXEMPLO 2-13

Um operador está digitando um programa em BASIC no teclado de um certo microcomputador. O computador converte cada tecla digitada para o código ASCII e armazena o código como um byte na memória. Determine as cadeias de bits que serão armazenadas na memória quando o operador digitar o seguinte comando BASIC:

GOTO 25

Solução

Localize cada caracter (incluindo o espaço) na Tabela 2-4 e registre seu código ASCII.

G	01000111
O	01001111
T	01010100
O	01001111
(espaço)	00100000
2	00110010
5	00110101

Observe que um 0 foi adicionado para o bit mais à esquerda de cada código ASCII porque os códigos devem ser armazenados como bytes (oito bits). Este acréscimo de um bit extra é chamado *preenchimento com 0s*.

Questões de Revisão

1. Codifique a seguinte mensagem em código ASCII usando a representação hexa:

"COST = $72."

2. A seguinte mensagem em código ASCII preenchido está armazenada em posições de memória consecutivas em um computador:

01010011 01010100 01001111 01010000

Qual é a mensagem?

2-9 MÉTODO DA PARIDADE PARA DETECÇÃO DE ERROS

A movimentação de dados binários e de códigos de um lugar para outro é a operação mais freqüentemente realizada em sistemas digitais. Aqui estão alguns exemplos:

- A transmissão de voz digitalizada através de um enlace de microondas
- A gravação e recuperação de dados de dispositivos de memória externa como fitas e discos magnéticos
- A transmissão de informação de um computador para um terminal de um usuário remoto ou para outro computador através das linhas telefônicas (usando um modem)

Sempre que uma informação é transmitida de um dispositivo (o transmissor) para outro dispositivo (o receptor), existe a possibilidade de que erros ocorram de modo que o receptor não receba a informação idêntica àquela que foi enviada pelo transmissor. A causa principal de erros de transmissão são *ruídos elétricos*, que consistem em flutuações espúrias de tensão ou corrente que estão presentes em diferentes graus em todos os sistemas eletrônicos. A Fig. 2-2 é uma ilustração simples de um tipo de erro de transmissão.

O transmissor envia um sinal digital serial relativamente livre de ruídos através de uma linha de sinal para o receptor. Entretanto, quando o sinal atinge o receptor, ele contém um certo nível de ruído sobreposto ao sinal original. Ocasionalmente, o ruído é grande o suficiente em amplitude e altera o nível lógico do sinal como acontece no ponto *x*. Quando isso ocorre, o receptor pode interpretar incorretamente aquele bit como 1 lógico, que não foi o que o transmissor enviou.

A maioria dos equipamentos digitais modernos é projetada para ser relativamente livre de erros, e a probabilidade de erros como mostrado na Fig. 2-2 é muito baixa. Entretanto, devemos compreender que sistemas digitais freqüentemente transmitem milhares, ou mesmo milhões, de bits por segundo, e assim mesmo uma taxa de ocorrência de erros muito baixa pode produzir um erro ocasional que pode ser um incômodo, ou mesmo um desastre. Por essa razão, muitos sistemas digitais empregam algum método para detecção (e algumas vezes correção) de erros. Um dos esquemas mais simples e mais amplamente usados para a detecção de erros é o **método da paridade.**

Bit de Paridade

Um **bit de paridade** é um bit extra que é anexado ao grupo de bits do código que está sendo transferido de um lugar para outro. O bit de paridade é 0 ou 1, dependendo do número de 1s contido no grupo. Dois métodos diferentes são usados.

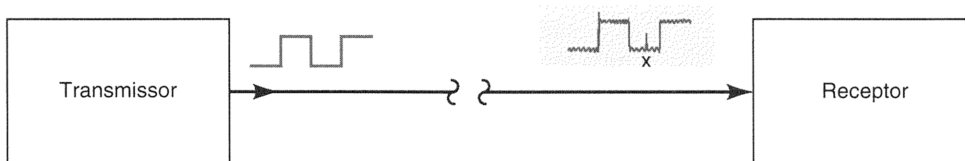


Fig. 2-2 Exemplo de ruído causando um erro na transmissão de dados digitais.

No método da *paridade par*, o valor do bit de paridade é escolhido de tal modo que o número total de 1s no grupo de bits do código (incluindo o bit de paridade) seja um número *par*. Por exemplo, suponha que o grupo é 1000011. Este é o caracter "C" em ASCII. Este grupo possui *três* 1s. Portanto, adicionamos um bit de paridade de 1 para fazermos o número total de 1s um número par. O *novo* grupo de bits do código, incluindo o bit de paridade, torna-se

1 1 0 0 0 0 1 1
↑——— bit de paridade adicionado*

Se o grupo de bits do código contém inicialmente um número par de 1s, o bit de paridade assume o valor 0. Por exemplo, se o grupo for 1000001 (o código ASCII para "A"), o bit de paridade deve ser 0, e o novo código, incluindo o bit de paridade, deve ser 01000001.

O método da *paridade ímpar* é usado exatamente do mesmo modo, com exceção de que o bit de paridade é escolhido de tal maneira que o número total de 1s (incluindo o bit de paridade) seja um número *ímpar*. Por exemplo, para o grupo 1000001, o bit de paridade deve ser 1. Para o grupo 1000011, o bit de paridade deve ser 0.

Independentemente de ser usada paridade par ou paridade ímpar, o bit de paridade torna-se parte integrante da palavra de código. Por exemplo, adicionando-se um bit de paridade ao código ASCII de sete bits, produz-se um código de oito bits. Assim, o bit de paridade é tratado como qualquer outro bit no código.

O bit de paridade é usado para detectar qualquer erro de *apenas um bit* que ocorra durante a transmissão de um código de um lugar para outro (e. g., de um computador para um terminal de vídeo). Por exemplo, suponha que o caracter "A" esteja sendo transmitido e que a paridade *ímpar* esteja sendo usada. O código transmitido deveria ser

1 1 0 0 0 0 0 1

Quando o circuito receptor receber esse código, ele verificará que o código contém um número ímpar de 1s (incluindo o bit de paridade). Se for este o caso, o receptor suporá que o código foi recebido corretamente. Agora, suponha que devido a algum ruído ou mau funcionamento o receptor receba o seguinte código:

1 1 0 0 0 0 0 0

O receptor constatará que esse código tem um número *par* de 1s. Isto revela ao receptor que deve haver um erro no código, já que presumivelmente o transmissor e o receptor tinham concordado em usar paridade ímpar. Não existe maneira, entretanto, de o receptor indicar qual bit está com erro, já que ele não sabe que código deveria ser.

*O bit de paridade pode ser colocado no início ou no fim do grupo, mas normalmente é colocado à esquerda do MSB.

Deveria ficar claro que este método da paridade não funcionaria se *dois* bits estivessem errados, porque dois erros não mudariam a paridade par ou ímpar do código. Na prática, o método da paridade é usado apenas em situações em que a probabilidade de erros simples é muito baixa e a probabilidade de erros duplos é essencialmente zero.

Quando o método da paridade está sendo usado, o transmissor e o receptor devem concordar, antecipadamente, se será usada a paridade par ou a paridade ímpar. Não existe vantagem de uma sobre a outra, embora a paridade par pareça ser usada mais freqüentemente. O transmissor deve anexar um bit de paridade apropriado para cada unidade de informação que transmite. Por exemplo, se o transmissor está enviando dados codificados em ASCII, ele anexa um bit de paridade para cada grupo do código ASCII de sete bits. Quando o receptor examinar os dados que recebeu do transmissor, ele verificará cada grupo de bits do código para constatar que o número total de 1s (incluindo o bit de paridade) é compatível com o tipo de paridade acordado previamente. Isto é comumente denominado de *verificação da paridade* dos dados. Na circunstância de ele detectar um erro, o receptor pode enviar uma mensagem de volta ao transmissor solicitando a retransmissão do último conjunto de dados. O procedimento exato que é seguido quando um erro é detectado dependerá do projeto do sistema em particular.

EXEMPLO 2-14

Computadores freqüentemente se comunicam com outros computadores remotos através de linhas telefônicas. Por exemplo, é assim que a comunicação pela Internet acontece. Quando um computador está transmitindo uma mensagem para outro, a informação é usualmente codificada em ASCII. Qual é a cadeia de bits real que um computador transmite para enviar a mensagem HELLO, usando ASCII com paridade par?

Solução

Primeiro procure o código ASCII para cada caracter da mensagem. Então para cada código conte o número de 1s. Se for um número par, anexe um 0 como MSB. Se for um número ímpar, anexe um 1. Assim, todos os códigos de oito bits (bytes) resultantes terão um número par de 1s (incluindo a paridade).

	bits de paridade par anexados
H-	0 1 0 0 1 0 0 0
E-	1 1 0 0 0 1 0 1
L-	1 1 0 0 1 1 0 0
L-	1 1 0 0 1 1 0 0
O-	1 1 0 0 1 1 1 1

Questões de Revisão

1. Anexe o bit de paridade ímpar para o código ASCII do símbolo $ e expresse o resultado em hexadecimal.
2. Anexe o bit de paridade par ao código BCD do número decimal 69.
3. Por que o método da paridade não pode detectar um erro duplo nos dados transmitidos?

2-10 REVISÃO

A título de revisão, aqui estão alguns exemplos a mais para ilustrar as operações apresentadas neste capítulo.

EXEMPLO 2-15

(a) Converta o decimal 135 para binário.

$$\frac{135}{2} = 67 + R1$$
$$\frac{67}{2} = 33 + R1$$
$$\frac{33}{2} = 16 + R1$$
$$\frac{16}{2} = 8 + R0$$
$$\frac{8}{2} = 4 + R0$$
$$\frac{4}{2} = 2 + R0$$
$$\frac{2}{2} = 1 + R0$$
$$\frac{1}{2} = 0 + R1$$
$$1\ 0\ 0\ 0\ 0\ 1\ 1\ 1_2$$

(b) Converta o decimal 76 para octal.

$$\frac{76}{8} = 9 + R4$$
$$\frac{9}{8} = 1 + R1$$
$$\frac{1}{8} = 0 + R1$$
$$1\ 1\ 4_8$$

(c) Converta o decimal 541 para hexadecimal.

$$\frac{541}{16} = 33 + R13$$
$$\frac{33}{16} = 2 + R1$$
$$\frac{2}{16} = 0 + R2$$
$$2\ 1\ D_{16}$$

(d) Converta o decimal 479 para BCD.

$$4\quad 7\quad 9$$
$$\downarrow\quad \downarrow\quad \downarrow$$
$$0100\ 0111\ 1001\quad \text{BCD}$$

(e) Converta o binário 101101 para decimal.

$$101101_2 = 1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0$$
$$= 32 + 8 + 4 + 1$$
$$= 45_{10}$$

(f) Converta o octal 6254 para decimal.

$$6254_8 = 6 \times 8^3 + 2 \times 8^2 + 5 \times 8^1 + 4 \times 8^0$$
$$= 6 \times 512 + 2 \times 64 + 5 \times 8 + 4 \times 1 = 3244_{10}$$

(g) Converta o hexa 1A3F para decimal.

$$1A3F = 1 \times 16^3 + 10 \times 16^2 + 3 \times 16^1 + 15 \times 16^0$$
$$= 4096 + 2560 + 48 + 15 = 6719_{10}$$

(h) Converta 010010010110 (BCD) para decimal.

$$0100\ 1001\ 0110\quad (\text{BCD})$$
$$4\quad 9\quad 6_{10}$$

(i) Converta o binário 10110111 para octal e para hexa.

$$010\ 110\ 111$$
$$2\quad 6\quad 7_8$$
$$1011\ 0111$$
$$B\quad 7_{16}$$

(j) Converta o hexadecimal E61 para binário.

$$E\quad 6\quad 1$$
$$\downarrow\quad \downarrow\quad \downarrow$$
$$1110\ 0110\ 0001_2$$

(k) Converta o octal 724 para binário.

$$7\quad 2\quad 4$$
$$\downarrow\quad \downarrow\quad \downarrow$$
$$111\ 010\ 100_2$$

(l) Adicione o bit de paridade ímpar no código ASCII de "Z".

Da Tabela 2-4, o código para "Z" é 1011010. O número de 1s neste grupo é quatro, um número par. Portanto, para achar a paridade ímpar, temos que anexar um 1 como bit de paridade (MSB) como segue:

1 1011010

Observe que o grupo de bits do código completo — incluindo o bit de paridade — agora tem um número ímpar de 1s.

RESUMO

1. Os sistemas de numeração octal e hexadecimal são usados em sistemas digitais e computadores como modos eficientes de representar quantidades binárias.
2. Em conversões entre octal e binário, um dígito octal corresponde a três bits. Em conversões entre hexa e binário, cada dígito hexa corresponde a quatro bits.
3. O método das divisões sucessivas é usado para converter números decimais para binário, octal ou hexadecimal.
4. Usando um número binário de *N* bits, podemos representar valores decimais de 0 até $2^N - 1$.
5. O código BCD para um número decimal é formado convertendo-se cada dígito do número decimal para o seu equivalente binário de quatro bits.
6. Um byte é uma cadeia de oito bits.
7. Um código alfanumérico usa grupos de bits para representar todos os vários caracteres e funções que fazem parte de um típico teclado de computador. O código ASCII é o código alfanumérico mais amplamente usado.
8. O método da paridade para detecção de erros anexa um bit de paridade especial para cada grupo de bits transmitido.

TERMOS IMPORTANTES*

sistema de numeração octal
sistema de numeração hexadecimal
codificação binária pura
decimal codificado em binário (código BCD)
byte
código alfanumérico
American Standard Code for Information Interchange (ASCII)
método da paridade
bit de paridade

PROBLEMAS

SEÇÕES 2-1 E 2-2

2-1. Converta os seguintes números binários para decimal.
(a) 10110 (d) 1111010111
(b) 10001101 (e) 10111111
(c) 100100001001

2-2. Converta os seguintes valores decimais para binário.
(a) 37 (d) 205
(b) 14 (e) 2313
(c) 189 (f) 511

2-3. Qual é o maior valor decimal que pode ser representado por um número binário de oito bits? E por um de 16 bits?

SEÇÃO 2-3

2-4. Converta cada número octal para seu equivalente decimal.
(a) 743 (d) 257
(b) 36 (e) 1204
(c) 3777

2-5. Converta cada um dos seguintes números decimais para octal.
(a) 59 (d) 65.536
(b) 372 (e) 255
(c) 919

2-6. Converta cada um dos valores octais do Problema 2-4 para binário.

2-7. Converta os números binários do Problema 2-1 para octal.

2-8. Relacione os números octais em seqüência desde 165_8 até 200_8.

2-9. Quando um número decimal grande tem que ser convertido para binário, às vezes é mais fácil convertê-lo primeiro para octal e depois de octal para binário. Tente este procedimento para 2313_{10} e compare com o procedimento usado no Problema 2-2 (e).

2-10. Quantos dígitos octais são necessários para representar números decimais até 20.000?

SEÇÃO 2-4

2-11. Converta os seguintes valores hexadecimais em decimal.
(a) 92 (d) 2C0
(b) 1A6 (e) 7FF
(c) 37FD

2-12. Converta os seguintes valores decimais para hexa.
(a) 75 (d) 25.619
(b) 314 (e) 4095
(c) 2048

2-13. Converta os números binários do Problema 2-1 para hexadecimal.

2-14. Converta os valores em hexa do Problema 2-11 para binário.

2-15. Liste os números hexadecimais em seqüência desde 280 até 2A0.

2-16. Quantos dígitos hexadecimais são necessários para representar números decimais até um milhão?

SEÇÃO 2-5

2-17. Codifique estes números decimais em BCD.
(a) 47 (d) 42.689.627
(b) 962 (e) 1204
(c) 187

2-18. Quantos bits são necessários para representar os números decimais na faixa de 0 até 999 usando-se a codificação binária pura? E usando o código BCD?

2-19. Os números seguintes estão em BCD. Converta-os para decimal.
(a) 1001011101010010 (c) 0111011101110101
(b) 000110000100 (d) 010010010010

SEÇÃO 2-7

2-20. (a) Quantos bits estão contidos em oito bytes?
(b) Qual é o maior número hexa que pode ser representado em quatro bytes?
(c) Qual é o maior valor decimal codificado em BCD que pode ser representado em três bytes?

SEÇÕES 2-8 E 2-9

2-21. Represente a instrução "X = 25/Y" no código ASCII (excluindo as aspas). Anexe o bit de paridade ímpar.

2-22. Anexe o bit de paridade *par* para cada um dos códigos ASCII do Problema 2-21 e forneça os resultados em hexa.

2-23. Os bytes a seguir (em hexadecimal) representam o nome de uma pessoa do modo como deveriam estar armazenados na memória de um computador. Cada byte é um código ASCII preenchido. Determine o nome da pessoa.

42 45 4E 20 53 4D 49 54 48

*Estes termos podem ser encontrados em negrito no capítulo e estão definidos no Glossário ao final do livro.

2-24. Converta os seguintes números decimais para o código BCD e depois anexe um bit de paridade *ímpar*.
(a) 74 (c) 165
(b) 38 (d) 9201

2-25. Em determinado sistema digital, os números decimais de 000 até 999 são representados no código BCD. Um bit de paridade *ímpar* também é incluído no fim de cada grupo de bits do código. Examine cada um dos grupos a seguir e admita que eles acabaram de ser transferidos de um lugar para outro. Alguns grupos contêm erros. Suponha que *não mais do que* dois erros ocorreram para cada grupo. Determine quais os grupos do código têm um erro apenas e quais deles têm *definitivamente* um erro duplo. (Sugestão: Lembre-se de que é utilizado o código BCD.)

(a) 1001010110000
↑ bit de paridade
(b) 0100011101100
(c) 0111110000011
(d) 1000011000101

2-26. Suponha que o receptor recebeu os seguintes dados a seguir do transmissor do Exemplo 2-14:

0 1 0 0 1 0 0 0
1 1 0 0 0 1 0 1
1 1 0 0 1 1 0 0
1 1 0 0 1 0 0 0
1 1 0 0 1 1 0 0

Que erros o receptor pode detectar nestes dados recebidos?

QUESTÕES DE FIXAÇÃO

2-27. Realize cada uma das conversões a seguir. Para algumas delas, você pode querer tentar vários métodos para ver qual deles é melhor para você. Por exemplo, uma conversão binário-decimal pode ser feita diretamente, ou pode ser feita como uma conversão binário-octal seguida de uma conversão octal-decimal.
(a) 1417_{10} = ______________ $_2$
(b) 255_{10} = ______________ $_2$
(c) 11010001_2 = ______________ $_{10}$
(d) 1110101000100111_2 = ______________ $_{10}$
(e) 2497_{10} = ______________ $_8$
(f) 511_{10} = ______________ $_8$
(g) 235_8 = ______________ $_{10}$
(h) 4316_8 = ______________ $_{10}$
(i) $7A9_{16}$ = ______________ $_{10}$
(j) $3E1C_{16}$ = ______________ $_{10}$
(k) 1600_{10} = ______________ $_{16}$
(l) 38.187_{10} = ______________ $_{16}$
(m) 865_{10} = ______________ (BCD)
(n) 100101000111 (BCD) = ______________ $_{10}$
(o) 465_8 = ______________ $_{16}$
(p) $B34_{16}$ = ______________ $_8$
(q) 01110100 (BCD) = ______________ $_2$
(r) 111010_2 = ______________ (BCD)

2-28. Represente o valor decimal 37 em cada um dos seguintes modos.
(a) binário puro
(b) BCD
(c) hexa
(d) ASCII (isto é, trate cada dígito como um caracter)
(e) octal

2-29. Complete os espaços com a(s) palavra(s) correta(s).
(a) A conversão de decimal para _______ requer sucessivas divisões por 8.
(b) A conversão de decimal para hexadecimal requer sucessivas divisões por ______.
(c) No código BCD, cada ________ é convertido para o seu binário equivalente de quatro bits.
(d) Um transmissor anexa um ___________ num grupo do código para permitir ao receptor detectar ___________.
(e) O código _______ é o código alfanumérico mais comum usado em sistemas computacionais.
(f) ____________ e __________ são freqüentemente utilizados como um modo conveniente de representar números binários grandes.
(g) Uma cadeia de oito bits é denominada um ______.

2-30. Escreva o número binário que resulta quando cada um dos números a seguir é incrementado de um.
(a) 0111
(b) 010000
(c) 1110

2-31. Repita o Problema 2-30 para a operação de decremento.

2-32. Escreva o número que resulta quando cada um dos valores a seguir é incrementado.
(a) 7777_8 (d) 2000_{16}
(b) 7777_{16} (e) $9FF_{16}$
(c) 2000_8 (f) 1000_{16}

2-33. Repita o Problema 2-32 para a operação de decremento.

EXERCÍCIOS DESAFIADORES

2-34. Em um microcomputador os *endereços* das posições de memória são números binários que identificam cada circuito de memória onde um byte está armazenado. O número de bits que compõem um endereço depende de quantas posições de memória existem. Como o número de bits pode ser muito grande, os endereços são freqüentemente especificados em hexa em vez de binário.
(a) Se um microcomputador usa um endereço de 20 bits, quantas posições de memória diferentes existem?
(b) Quantos dígitos hexadecimais são necessários para representar o endereço de uma posição de memória?
(c) Qual é o endereço hexa da 256ª posição de memória?
(*Nota*: o primeiro endereço é sempre 0.)

2-35. Em um CD de áudio, o sinal de tensão do áudio é geralmente amostrado cerca de 44.000 vezes por segundo, e o valor de cada amostra é gravado na superfície do CD como um número binário. Em outras palavras, cada número binário gravado representa o valor da tensão em um ponto da forma de onda do sinal de áudio.
(a) Se os números binários têm seis bits de tamanho, quantos valores diferentes de tensão podem ser representados por apenas um número binário? Repita para oito bits e para dez bits.
(b) Se números de dez bits são usados, quantos bits devem ser gravados num CD em 1 segundo?
(c) Se um CD pode geralmente armazenar 5 bilhões de bits, quantos segundos de áudio podem ser gravados quando números de dez bits são usados?

2-36. Uma câmera eletrônica em preto-e-branco constrói um reticulado sobre uma imagem e então mede e grava um número binário representando o nível de cinza que ela detecta em cada célula do reticulado. Por exemplo, se números de quatro bits são usados, o valor do preto é ajustado para 0000 e o valor do branco para 1111, e qualquer nível de cinza fica entre 0000 e 1111. Se números de seis bits são usados, o preto é 000000, o branco é 111111, e todos os cinzas estão entre eles.

Suponha que queiramos distinguir entre 254 níveis diferentes de cinza em cada célula do reticulado. Quantos bits teríamos que usar para representar esses níveis?

2-37. Construa uma tabela mostrando as representações binária, octal, hexadecimal e BCD para todos os números decimais de 0 até 15. Compare sua tabela com a Tabela 2-3.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 2-1

1. 2267
2. 32768

SEÇÃO 2-2

1. 1010011
2. 1011011001
3. 20 bits

SEÇÃO 2-3

1. 396
2. 222; 010010010
3. 235
4. 627, 630, 631
5. 1111001111
6. 699
7. 0 até 4095

SEÇÃO 2-4

1. 9422
2. C2D; 110000101101
3. 97B5
4. E9E, E9F, EA0, EA1
5. 757
6. 0 até 65.535

SEÇÃO 2-5

1. 10110010_2; 000101111000 (BCD)
2. 32
3. Vantagem: facilidade de conversão. Desvantagem: BCD requer mais bits.

SEÇÃO 2-7

1. Dois
2. 99

SEÇÃO 2-8

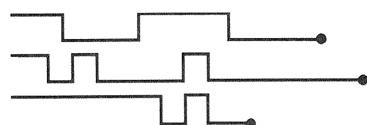
1. 43, 4F, 53, 54, 20, 3D, 20, 24, 37, 32
2. STOP

SEÇÃO 2-9

1. A4
2. 001101001
3. Dois erros nos dados não mudariam a paridade par ou ímpar dos dados.

CAPÍTULO 3

Portas Lógicas e Álgebra Booleana



■ SUMÁRIO

OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Realizar as três operações lógicas básicas.
- Descrever a operação das portas AND, NAND, OR, NOR e NOT (INVERSOR), bem como construir as tabelas-verdade para as mesmas.
- Desenhar diagramas de tempo para as várias portas lógicas.
- Escrever expressões booleanas para as portas lógicas e para combinações das mesmas.
- Implementar circuitos lógicos utilizando portas básicas AND, OR e NOT.
- Estimar o potencial da álgebra booleana para simplificar circuitos lógicos complexos.
- Utilizar os teoremas de DeMorgan para simplificar expressões lógicas.
- Utilizar uma das portas lógicas universais (NAND ou NOR) para implementar um circuito representado por uma expressão booleana.
- Explicar as vantagens de construir um diagrama de circuito lógico usando a representação alternativa de símbolos para as portas lógicas *versus* a representação padrão.
- Descrever o conceito de sinais lógicos ativos em nível BAIXO e ativos em nível ALTO.
- Desenhar e interpretar circuitos lógicos que utilizam a representação de portas lógicas padrão IEEE/ANSI.

INTRODUÇÃO

Como foi mencionado no Cap. 1, circuitos digitais (lógicos) operam de modo binário onde cada tensão de saída ou entrada tem o valor 0 ou 1. As designações 0 e 1 representam intervalos de tensão predefinidos. Esta característica dos circuitos digitais nos permite utilizar a **álgebra booleana*** como uma ferramenta de análise e projeto de circuitos digitais. A álgebra booleana é uma ferramenta matemática relativamente simples que nos permite descrever a relação entre a(s) saída(s) de um circuito lógico e suas entradas através de uma equação (expressão booleana). Neste capítulo estudaremos os circuitos lógicos mais elementares, as *portas lógicas*, que são os blocos fundamentais a partir dos quais todos os outros circuitos lógicos e sistemas digitais são construídos. Veremos como a operação das diferentes portas lógicas e de circuitos mais complexos, formados pela combinação de portas lógicas, pode ser descrita e analisada utilizando a álgebra booleana. Também vislumbraremos como a álgebra booleana pode ser usada para simplificar a expressão booleana de um circuito, de modo a permitir que este circuito possa ser reconstruído utilizando um menor número de portas lógicas e/ou de conexões entre estas. Muito mais será dito sobre simplificação de circuitos no Cap. 4.

A álgebra booleana é também uma ferramenta valiosa para projetar um circuito que produzirá a relação desejada entre a entrada e a saída. Introduziremos a idéia básica neste capítulo e, depois, faremos uma cobertura mais completa deste tópico quando estudarmos o projeto de circuitos lógicos no Cap. 4.

Como a álgebra booleana expressa a operação de circuitos lógicos de forma algébrica, ela se apresenta como a forma ideal de descrever a operação de um circuito lógico para um programa de computador que necessite de informações sobre o circuito em questão. Este programa pode ser um procedimento de simplificação de circuitos, que recebe como entrada a equação em álgebra booleana, simplifica-a e fornece como saída uma versão simplificada do circuito lógico original. Uma outra aplicação possível para esse programa seria a geração de *fuse maps* (mapas de fusíveis) necessários para a programação de um dispositivo de lógica programável (*PLD — Programmable Logic Device*). O operador forneceria como entrada as equações booleanas que representariam a operação desejada do circuito e o programa as converteria em *fuse maps*. Estudaremos este processo em detalhe no Cap. 12.

Sem dúvida, a álgebra booleana é uma ferramenta muito valiosa para descrever, projetar e implementar circuitos digitais. O estudante que deseja atuar na área digital é estimulado a trabalhar bastante para compreender a lógica booleana e sentir-se à vontade com ela (acredite, ela é muito, muito mais fácil que a álgebra convencional). Faça *todos* os exemplos, exercícios e problemas, mesmo aqueles que seu professor não tiver recomendado. Quando eles acabarem, faça os seus próprios. O tempo gasto terá valido a pena quando você observar que suas habilidades aumentam e sua confiança cresce.

3-1 CONSTANTES E VARIÁVEIS BOOLEANAS

A álgebra booleana possui uma diferença fundamental em relação à álgebra convencional. Na álgebra booleana, constantes e variáveis possuem apenas dois valores permitidos, 0 ou 1. Uma variável booleana é uma quantidade que pode, em momentos diferentes, ser igual a 0 ou 1. Variáveis booleanas são geralmente utilizadas para representar o nível de tensão presente nas ligações ou nos terminais de entrada/saída do circuito. Por exemplo, em um certo sistema digital, o valor booleano 0 é dado para qualquer nível de tensão situado no intervalo entre 0 e 0,8 V, enquanto o valor booleano 1 é dado para qualquer nível de tensão situado no intervalo entre 2 a 5 V.*

Assim, 0 e 1 booleanos não são números de fato, mas, ao contrário, representam o estado do nível de tensão de

*A álgebra booleana foi desenvolvida pelo matemático George Boole (1815-1864) para o estudo da lógica. Ela foi apresentada em 1854 no trabalho intitulado *An Investigation of the Laws of Thought*. (N. T.)

*Níveis de tensão entre 0,8 e 2 V são indefinidos (não são 0 nem 1) e em circunstâncias normais não devem ocorrer.

TABELA 3-1

Nível lógico 0	Nível lógico 1
Falso	Verdadeiro
Desligado	Ligado
Baixo	Alto
Não	Sim
Chave aberta	Chave fechada

uma variável, ou, como é chamado, o seu **nível lógico.** Diz-se que o nível de tensão em um circuito digital está no nível lógico 0 ou no nível lógico 1, dependendo do seu valor numérico de fato. Em lógica digital, vários outros termos são usados como sinônimos de 0 e 1. Alguns dos mais comuns são mostrados na Tabela 3-1. Usaremos as designações 0/1 e BAIXO/ALTO na maioria das vezes.

Conforme dissemos na introdução, a álgebra booleana é um modo de expressar a relação entre as entradas e as saídas de um circuito lógico. As entradas são consideradas variáveis lógicas cujos níveis lógicos determinam, a qualquer momento, os níveis lógicos da saída. A partir de agora, utilizaremos letras para representar variáveis lógicas. Por exemplo, A poderia representar uma certa entrada ou saída de um circuito digital, e em qualquer instante necessariamente teríamos ou $A = 0$ ou $A = 1$.

Como apenas dois valores são possíveis, a álgebra booleana é relativamente mais fácil de se trabalhar do que a álgebra convencional. Na álgebra booleana não existem frações, decimais, números negativos, raízes quadradas, raízes cúbicas, logaritmos, números imaginários e assim por diante. Na verdade, na álgebra booleana existem apenas *três* operações básicas: *OR* (OU), *AND* (E) e *NOT* (NÃO).

Essas operações básicas são chamadas *operações lógicas*. Circuitos digitais chamados *portas lógicas* podem ser construídos a partir de diodos, transistores e resistores conectados de um modo pelo qual a saída do circuito seja o resultado da operação lógica básica (*OR, AND, NOT*) realizada sobre suas entradas. Utilizaremos a álgebra, primeiramente para descrever e analisar essas portas lógicas básicas, e posteriormente para analisar e projetar combinações dessas portas lógicas conectadas como circuitos lógicos.

3-2 TABELAS-VERDADE

A **tabela-verdade** é uma maneira de descrever como a saída de um circuito lógico depende dos níveis lógicos presentes nas entradas do circuito. A Fig. 3-1(a) mostra a tabela-verdade para um tipo de circuito lógico de duas entradas. A tabela relaciona todas as combinações possíveis dos níveis lógicos presentes nas entradas A e B com o nível correspondente da saída x. A primeira linha da tabela mostra que quando A e B estão ambos em nível 0, a saída x está no nível 1, ou, de modo equivalente, no estado 1. A segunda linha da tabela mostra que quando a entrada B muda para o estado 1, de modo que $A = 0$ e $B = 1$, a saída x torna-se 0. De maneira similar, a tabela mostra o que acontece com o estado da saída para qualquer conjunto de condições de entrada.

As Figs. 3-1(b) e (c) mostram exemplos de tabelas-verdade para circuitos de três e de quatro entradas. Novamente, cada tabela enumera todas as combinações possíveis dos níveis lógicos de entrada na esquerda, juntamente com o nível lógico resultante para a saída x na direita. É claro que o valor real de x dependerá do tipo de circuito lógico utilizado.

Observe que existem 4 linhas para um tabela-verdade de duas entradas, 8 linhas para uma tabela-verdade de três entradas, e 16 linhas para uma tabela de quatro entradas. O número de combinações de entrada será igual a 2^N para uma tabela-verdade de N entradas. Note também que a lista de todas as combinações possíveis de entrada acompanha a seqüência de contagem binária, e, assim, torna-se bastante simples escrever todas as combinações possíveis sem esquecer nenhuma.

Questões de Revisão

1. Qual é o estado da saída para o circuito de quatros entradas representado na Fig. 3-1(c), quando todas as entradas forem iguais a 1?
2. Repita a questão 1 para as seguintes condições de entrada: $A = 1$, $B = 0$, $C = 1$, $D = 0$.
3. Quantas linhas deve ter uma tabela para representar um circuito de cinco entradas?

Entradas — Saída

A	B	x
0	0	1
0	1	0
1	0	1
1	1	0

A, B → ? → x

(a)

A	B	C	x
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

(b)

A	B	C	D	x
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

(c)

Fig. 3-1 Exemplos de tabelas-verdade para circuitos (a) de duas entradas, (b) de três entradas e (c) de quatro entradas.

3-3 OPERAÇÃO OR COM PORTAS OR

A **operação OR** é a primeira das três operações booleanas básicas a ser estudada. A tabela-verdade na Fig. 3-2(a) mostra o que acontece quando duas entradas lógicas, *A* e *B*, são combinadas através da operação OR para produzir a saída *x*. A tabela mostra que *x* é igual a 1 para todas as combinações dos níveis de entrada onde uma *ou* mais entradas são iguais a 1. O único caso onde *x* é igual a 0 ocorre quando todas as entradas são iguais a 0.

A expressão booleana para a operação OR é dada por:

$$x = A + B$$

Nesta expressão, o sinal de + não representa a operação de adição ordinária, mas representa a operação OR. A operação OR é semelhante à adição ordinária, exceto para o caso em que *A* e *B* são ambos iguais a 1. Neste caso, a operação OR produz 1 + 1 = 1, e não 1 + 1 = 2, como seria no caso de uma adição. Na álgebra booleana, 1 é o valor máximo que pode ser obtido, e assim nunca poderemos ter um resultado maior do que 1. Essa afirmação continua sendo verdadeira quando combinamos três entradas utilizando a operação OR. Aqui teremos $x = A + B + C$. Se considerarmos o caso em que todas as três entradas são iguais a 1:

$$x = 1 + 1 + 1 = 1$$

Novamente, o resultado da operação OR, quando mais de uma entrada é igual a 1, é *sempre* igual a 1.

A expressão lógica $x = A + B$ é lida como "*x* é igual a *A* OR *B*". O mais importante a ser lembrado é que o sinal de +, que aparece na expressão, representa a operação OR que foi definida através da tabela-verdade na Fig. 3-2(a), e não a operação de adição ordinária.

Porta OR

Em circuitos digitais, uma **porta OR*** é um circuito que possui duas ou mais entradas e cuja saída é igual à combinação das entradas através da operação OR. A Fig. 3-2(b) mostra o símbolo para uma porta OR de duas entradas. As entradas *A* e *B* são níveis lógicos de tensão, e a saída *x* é um nível lógico de tensão cujo valor é o resultado da operação OR sobre as entradas *A* e *B*, isto é, $x = A + B$. Em outras palavras, a porta OR funciona de tal modo que sua saída será ALTA (nível lógico 1) se *A ou B ou ambas* forem iguais a 1. A saída da porta OR será BAIXA (nível lógico 0) apenas se todas as entradas forem iguais a 0.

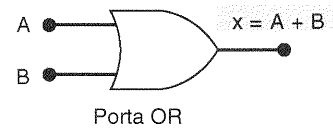
Esta mesma idéia pode ser estendida para um maior número de entradas. A Fig. 3-3 mostra uma porta OR de 3 entradas e sua tabela-verdade. O exame desta tabela-verdade mostra novamente que a saída será igual a 1 para todos os casos nos quais uma ou mais entradas são iguais a 1. Este princípio geral é o mesmo para portas OR com qualquer número de entradas.

Usando a linguagem da álgebra booleana, a saída *x* pode ser expressa como $x = A + B + C$, onde novamente deve-

OR

A	B	x = A + B
0	0	0
0	1	1
1	0	1
1	1	1

(a)



Porta OR

(b)

Fig. 3-2 (a) Tabela-verdade que define a operação OR; (b) símbolo para uma porta OR de duas entradas.

mos enfatizar que o sinal de + representa a operação OR. A saída de qualquer porta OR, então, pode ser expressa pela combinação das entradas através da operação OR. Utilizaremos este fato quando estivermos analisando circuitos lógicos.

Resumo da Operação OR

Os pontos mais importantes a serem lembrados no que se refere à operação OR e às portas OR são:

1. A operação OR produz 1 como resultado, quando *qualquer* uma das variáveis for igual a 1.
2. A operação OR produz 0 como resultado, quando todas as variáveis forem iguais a 0.
3. Na operação OR, 1 + 1 = 1, 1 + 1 + 1 = 1, e assim por diante.
4. A porta OR é um circuito lógico que realiza a operação OR sobre as entradas lógicas do circuito.

EXEMPLO 3-1

Em muitos sistemas de controle industriais é necessário ativar uma função de saída sempre que uma das várias entradas for ativada. Por exemplo, em um processo químico, pode ser desejável que um alarme seja ativado toda vez que a temperatura do processo exceder um valor máximo *ou* sempre que a pressão estiver acima de um certo limite. A Fig. 3-4 mostra um diagrama de blocos desta situação. O circuito transdutor de temperatura produz uma tensão proporcional à temperatura do processo. Esta tensão, V_T, é comparada com uma tensão de referência de temperatura, V_{TR}, através de um circuito comparador. A saída do comparador está normalmente

A
B
C
x = A + B + C

A	B	C	x = A + B + C
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Fig. 3-3 Símbolo e a tabela-verdade para uma porta OR de três entradas.

*O termo porta [do inglês, *gate*] deriva da operação de habilitar/inibir a ser discutida no Cap. 4.

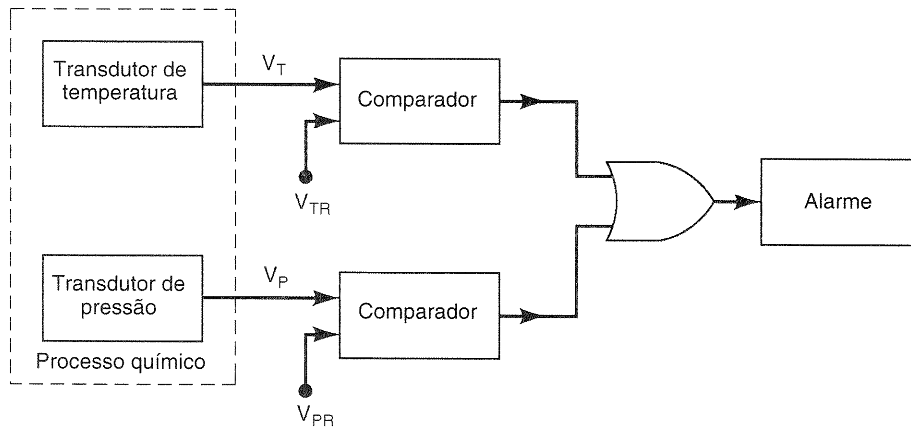


Fig. 3-4 Exemplo de utilização da porta OR em um sistema de alarme.

com uma tensão baixa (nível lógico 0), mas esta muda para uma tensão alta (nível lógico 1) quando V_T excede V_{TR}, indicando que a temperatura do processo é excessiva. Um arranjo similar é feito para a medição da pressão, de modo que a saída do respectivo comparador passa do nível baixo para o alto quando a pressão for excessiva.

Uma vez que desejamos que o alarme seja ativado quando *ou* a temperatura *ou* a pressão seja muito alta, podemos conectar as saídas dos comparadores a uma porta OR de duas entradas. A saída da porta OR será ALTA (1) para qualquer uma das condições de alarme, fazendo com que o mesmo seja ativado. Esta mesma idéia pode ser obviamente estendida para situações com mais do que duas variáveis de processo.

EXEMPLO 3-2

Determine a saída da porta OR mostrada na Fig. 3-5. As entradas da porta OR são *A* e *B* que variam segundo o diagrama de tempo apresentado. Por exemplo, *A* começa em BAIXO em t_0, passa para ALTO em t_1, e retorna a BAIXO em t_3, e assim por diante.

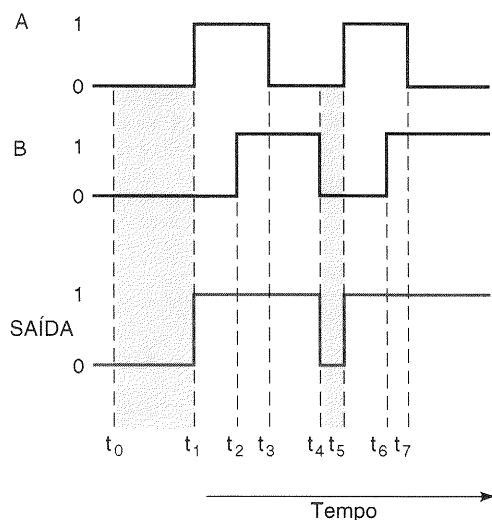


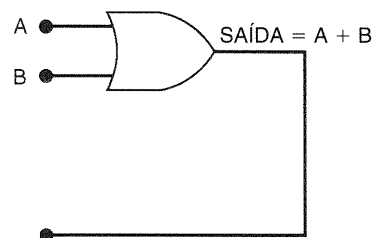
Fig. 3-5 Exemplo 3-2.

Solução

A saída da porta OR pode ser determinada observando-se que ela estará em ALTO sempre que *qualquer* uma das entradas estiver em nível alto. Quando *A* passa para ALTO em t_1, SAÍDA passará para ALTO. A SAÍDA permanecerá em ALTO até t_4, quando ambas as entradas estarão em BAIXO. Observe que as mudanças nos níveis lógicos das entradas que ocorrem em t_2 e t_3 não têm efeito na SAÍDA, uma vez que uma das entradas permanece em nível ALTO enquanto a outra está mudando. Enquanto uma das entradas da porta OR estiver em ALTO, a saída permanecerá em ALTO, não importando o que estiver acontecendo nas outras entradas. Este mesmo raciocínio pode ser usado para determinar o restante do diagrama de tempo para SAÍDA.

EXEMPLO 3-3A

Para o exemplo mostrado na Fig. 3-6, determine a forma de onda na saída da porta OR.



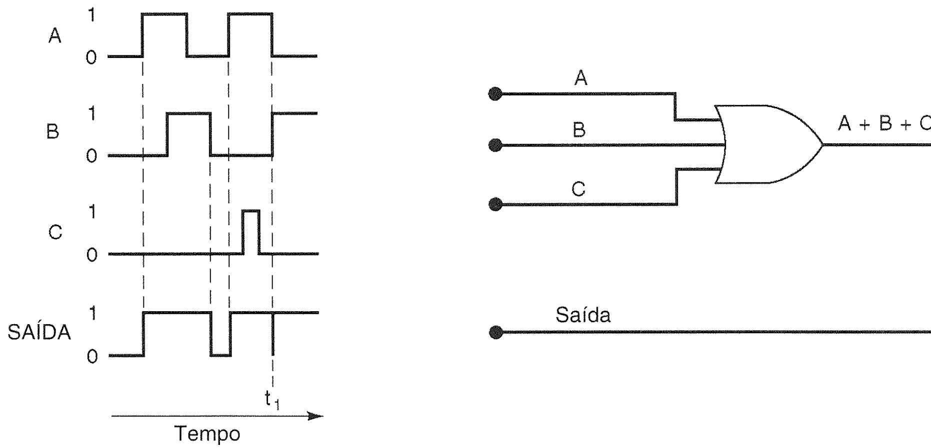


Fig. 3-6 Exemplos 3-3A e 3-3B.

Solução

As três entradas da porta OR, *A*, *B* e *C*, estão variando, conforme as formas de onda mostradas no diagrama. A saída da porta OR será determinada observando que esta será alta sempre que *qualquer* uma das três entradas estiver em nível alto. Usando este raciocínio, a forma de onda da saída da porta OR é apresentada na figura. Devemos prestar bastante atenção no que acontece no instante t_1. O diagrama mostra que neste instante de tempo a entrada *A* está passando de alto para baixo, enquanto a entrada *B* está passando de baixo para alto. Como estas entradas estão fazendo suas transições aproximadamente no mesmo instante, e como essas transições duram um certo tempo, existe um pequeno intervalo em que ambas as entradas dessa porta OR estão na faixa indefinida entre 0 e 1. Quando isso ocorre, a saída da porta OR também possui um valor situado nesse intervalo indefinido, caracterizado por um pulso espúrio e estreito (*glitch* ou *spike*) na forma de onda da saída em t_1. A ocorrência do *glitch*, sua amplitude e largura irão depender da velocidade com que as transições acontecem.

EXEMPLO 3-3B

O que aconteceria ao *glitch* mostrado na Fig. 3-6 caso a entrada *C* permanecesse em nível ALTO enquanto *A* e *B* estivessem mudando de estado em t_1?

Solução

Com a entrada *C* em ALTO no instante t_1, a saída da porta OR permanecerá em ALTO independentemente do que estiver ocorrendo nas outras entradas, porque se qualquer uma das entradas estiver em ALTO a saída permanecerá em ALTO, e portanto o *glitch* não aparecerá na saída.

Questões de Revisão

1. Qual é a única combinação de valores das entradas que produz um nível BAIXO na saída de qualquer porta OR?
2. Escreva a expressão booleana para uma porta OR de seis entradas.
3. Se a entrada *A* mostrada na Fig. 3-6 fosse mantida permanentemente em nível 1, qual seria a forma de onda resultante na saída?

3-4 OPERAÇÃO AND COM PORTAS AND

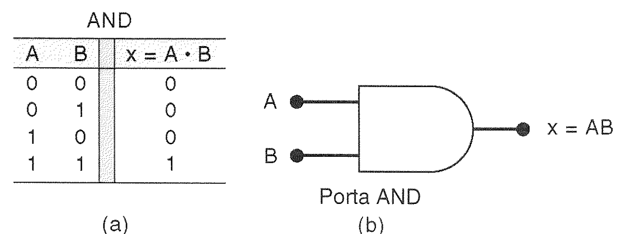
A **operação AND** é a segunda operação booleana básica. A tabela-verdade que aparece na Fig. 3-7(a) mostra o que acontece quando duas entradas lógicas, *A* e *B*, são combinadas usando a operação AND para produzir a saída *x*. A tabela mostra que *x* está em nível lógico 1 somente quando tanto *A* como *B* estão em nível lógico 1. Para qualquer outro caso, onde uma das entradas é 0, a saída é 0.

A expressão booleana para a operação AND é

$$x = A \cdot B$$

Nesta expressão, o sinal · expressa a operação AND, e não a multiplicação ordinária. Entretanto, a operação AND sobre variáveis booleanas opera da mesma maneira que a multiplicação ordinária, como pode ser visto através de um exame da tabela-verdade. Assim, podemos pensar nas duas operações como se fossem apenas uma. Essa característica pode ser de grande ajuda na análise de expressões lógicas que contenham operações AND.

A expressão $x = A \cdot B$ é lida como "x = A AND B". O sinal · é geralmente omitido de modo que a expressão se torna apenas $x = AB$. A coisa mais importante a ser lembrada é que a operação AND produzirá 1 como resultado *ape-*



AND

A	B	x = A • B
0	0	0
0	1	0
1	0	0
1	1	1

Fig. 3-7 (a) Tabela-verdade para a operação AND; (b) símbolo da porta AND.

A	B	C	x = ABC
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

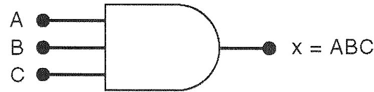


Fig. 3-8. Tabela-verdade e o símbolo para uma porta AND de três entradas.

nas quando todas as entradas (variáveis) forem iguais a 1, exatamente como na multiplicação. Este fato permanece verdadeiro para o caso de termos mais de duas entradas. Por exemplo, quando a operação AND é realizada sobre três entradas, temos $x = A \cdot B \cdot C = ABC$. O único momento em que x pode ser igual a 1 é quando $A = B = C = 1$.

Porta AND

O símbolo lógico para uma **porta AND** de duas entradas pode ser visto na Fig. 3-7(b). A saída da porta AND é igual ao produto das entradas lógicas, isto é, $x = AB$. Em outras palavras, a porta AND é um circuito que opera de tal maneira que sua saída está em ALTO apenas quando todas as entradas estão em ALTO. Para todos os outros casos, a saída da porta estará em BAIXO.

Esse mesmo modo de operação é característico em portas AND com mais de duas entradas. Por exemplo, uma porta AND de três entradas e a tabela-verdade correspondente podem ser vistas na Fig. 3-8. Mais uma vez, observe que a saída da porta é 1 apenas para o caso em que $A = B = C = 1$. A expressão para a saída é $x = ABC$. Para o caso de uma porta AND de quatro entradas, a expressão é $x = ABCD$, e assim por diante.

Observe a diferença entre os símbolos das portas AND e OR. Sempre que você vir o símbolo de uma porta AND em um diagrama de circuitos lógicos, isto lhe diz que a saída estará em ALTO *apenas* quando *todas* as entradas estiverem em ALTO. Sempre que você vir o símbolo de uma porta OR, isto significa que a saída estará em ALTO quando *qualquer* uma das entradas estiver em ALTO.

Resumo da Operação AND

1. A operação AND é realizada exatamente do mesmo modo que a multiplicação ordinária de 0s e 1s.
2. A saída é igual a 1 quando todas as entradas forem iguais a 1.
3. A saída é 0 para o caso em que uma ou mais entradas são iguais a 0.
4. Uma porta AND é um circuito lógico que realiza a operação AND nas entradas do circuito.

EXEMPLO 3-4

Determine a forma de onda da saída x da porta AND mostrada na Fig. 3-9, dadas as formas de onda das entradas.

Solução

A saída da porta AND é determinada observando que ela estará em ALTO apenas quando todas as entradas estiverem em ALTO ao mesmo tempo. Para as formas de onda fornecidas, isto acontece apenas durante os intervalos t_2-t_3 e t_6-t_7. Em todos os outros intervalos, uma ou mais entradas estão em 0, produzindo portanto um nível BAIXO na saída. Observe que mudanças nos níveis de entrada que ocorrem enquanto uma das entradas está em nível BAIXO não têm efeito na saída.

EXEMPLO 3-5A

Determine a forma de onda da saída para a porta AND mostrada na Fig. 3-10.

Solução

A saída x será igual a 1 apenas quando A e B estiverem em ALTO ao mesmo tempo. A partir deste fato, podemos determinar a forma de onda de x como está mostrado na figura.

Observe que a forma de onda de x é igual a 0 toda vez que B é igual a 0, independentemente do que acontece com

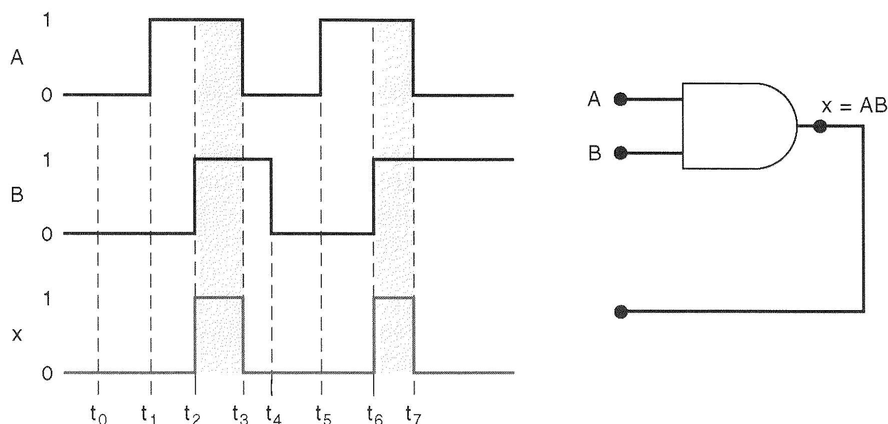


Fig. 3-9 Exemplo 3-4.

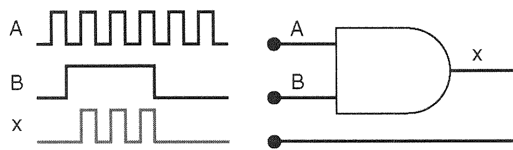
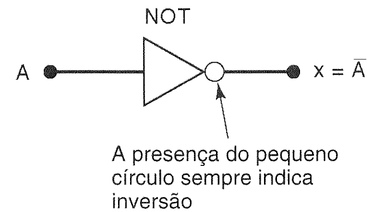


Fig. 3-10 Exemplos 3-5A e 3-5B.

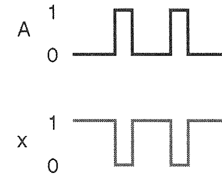
NOT

A	$x = \overline{A}$
0	1
1	0

(a)



(b)



(c)

Fig. 3-11 (a) Tabela-verdade; (b) símbolo para o INVERSOR (NOT); (c) formas de onda.

a entrada *A*. Também é importante notar que sempre que *B* é igual a 1 a forma de onda de *x* é igual à de *A*. Então podemos pensar na entrada *B* como uma entrada de *controle*, cujo nível lógico determina se a forma de onda de *A* chega ou não na saída *x*. Nesta situação, a porta AND é usada como um *circuito inibidor*. Podemos dizer que $B = 0$ é a condição de inibição que força que a saída seja igual a 0. Ao contrário, $B = 1$ é a condição de *habilitação*, que permite que *A* chegue até a saída. Esta operação inibidora é uma importante aplicação das portas AND que encontraremos mais tarde.

EXEMPLO 3-5B

O que acontecerá com a forma de onda da saída *x* na Fig. 3-10 se a entrada *B* permanecer em nível 0?

Solução

Enquanto *B* for mantido em BAIXO, a saída *x* também permanecerá em BAIXO. Podemos chegar a esta conclusão de dois modos: o primeiro seria observar que com $B = 0$ temos $x = A \cdot B = A \cdot 0 = 0$, uma vez que o resultado da operação AND (multiplicação), quando uma das entradas é 0, é sempre 0. O segundo modo seria observar que uma porta AND necessita que todas as suas entradas estejam em ALTO para que a saída seja ALTO, e isto não acontece quando *B* é mantido em BAIXO.

Questões de Revisão

1. Qual é a única combinação de entrada que irá produzir um nível ALTO na saída de uma porta AND de cinco entradas?
2. Qual é o nível lógico que deve ser aplicado na segunda entrada de uma porta AND de duas entradas para que o sinal aplicado na primeira entrada seja inibido (impedido) de chegar na saída?
3. *Falso* ou *verdadeiro*: A saída de uma porta AND sempre difere da saída de uma porta OR para as mesmas condições de entrada.

3-5 OPERAÇÃO NOT

A **operação NOT** é realizada, ao contrário das operações AND e OR, sobre uma única entrada. Por exemplo, se a variável *A* é sujeita à operação NOT, o resultado *x* pode ser expresso como:

$$x = \overline{A}$$

onde a barra sobreposta representa a operação NOT. Esta expressão é lida como "*x* é igual a NOT *A*" ou "*x* é igual ao *inverso* de *A*" ou "*x* é igual ao *complemento* de *A*".* Cada uma destas expressões é de uso comum, e todas indicam que o nível lógico de $x = \overline{A}$ é *oposto* ao valor lógico de *A*. A tabela-verdade mostrada na Fig. 3-11(a) esclarece esta afirmação para os dois casos possíveis, $A = 0$ e $A = 1$, isto é:

$$\overline{1} = 0 \quad \text{porque NOT 1 é 0}$$

e

$$\overline{0} = 1 \quad \text{porque NOT 0 é 1}$$

A operação NOT é também chamada de **inversão** ou **complemento;** estes termos serão usados de modo intercambiável durante o restante do livro. Apesar de sempre utilizarmos a barra sobreposta para representar inversão, é importante mencionar que um outro símbolo para representar a inversão é o apóstrofo ('), isto é:

$$A' = \overline{A}$$

Ambos os símbolos são reconhecidos como indicadores da operação de inversão.

Circuito NOT (INVERSOR)

A Fig. 3-11(b) mostra o símbolo para a representação do **circuito NOT**, que é mais comumente chamado de **INVERSOR.** Este circuito tem *sempre* uma única entrada, e o nível lógico de sua saída é sempre oposto ao nível lógico da entrada. A Fig. 3-11(c) mostra como o INVERSOR age sobre o sinal de entrada. Ele inverte (complementa) o sinal de entrada em todos os pontos da forma de onda da entrada.

---

*Usa-se também dizer "*x* é igual a *A* barrado". (*N. T.*)

Resumo das Operações Booleanas

As regras para as operações AND, OR e NOT podem ser resumidas como segue:

OR	AND	NOT
$0 + 0 = 0$	$0 \cdot 0 = 0$	$\overline{0} = 1$
$0 + 1 = 1$	$0 \cdot 1 = 0$	$\overline{1} = 0$
$1 + 0 = 1$	$1 \cdot 0 = 0$	
$1 + 1 = 1$	$1 \cdot 1 = 1$	

Questões de Revisão

1. A saída do INVERSOR da Fig. 3-11 é conectada à entrada de um segundo INVERSOR. Determine o nível lógico da saída do segundo INVERSOR para cada nível lógico da entrada *A*.
2. A saída da porta AND da Fig. 3-7 é conectada à entrada de um INVERSOR. Escreva a tabela-verdade que relaciona a saída *y* do INVERSOR com cada combinação das entradas *A* e *B*.

3-6 DESCREVENDO CIRCUITOS LÓGICOS ALGEBRICAMENTE

Qualquer circuito lógico, independentemente de sua complexidade, pode ser completamente descrito usando as operações booleanas previamente definidas, porque as portas AND, OR e NOT são os blocos básicos para a construção de sistemas digitais. Por exemplo, considere o circuito da Fig. 3-12. O circuito possui 3 entradas, *A*, *B* e *C*, e uma única saída, *x*. Utilizando as expressões booleanas para cada porta, podemos facilmente determinar a expressão para a saída.

A expressão para a saída da porta AND é escrita como $A \cdot B$. Esta saída é conectada a uma porta OR, juntamente com *C*, que é a outra entrada do circuito. A porta OR opera sobre as entradas de modo que a saída seja o resultado de uma operação OR sobre as entradas. Assim, podemos expressar a saída da porta OR como $x = A \cdot B + C$ (esta última expressão também poderia ter sido escrita como $x = C + A \cdot B$, uma vez que a ordem dos termos não importa na operação OR).

Ocasionalmente, pode haver dúvida em relação a qual operação deve ser realizada primeiro. A expressão $A \cdot B + C$ pode ser interpretada de dois modos: (1) é feita a operação $A \cdot B$ OR *C*, ou (2) é feita a operação *A* AND $B + C$. Para evitar essa confusão, fica definido que, caso uma expressão possua as operações AND e OR, as operações AND são realizadas primeiro, a não ser que existam *parênteses* na expressão, neste caso, a operação dentro dos parênteses é realizada primeiro. Esta é a mesma regra usada na álgebra comum para determinar a ordem das operações.

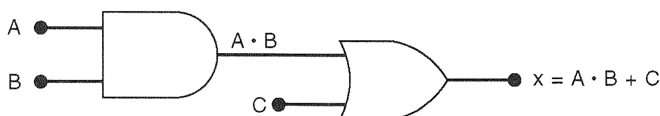


Fig 3-12 Circuito lógico com sua expressão booleana.

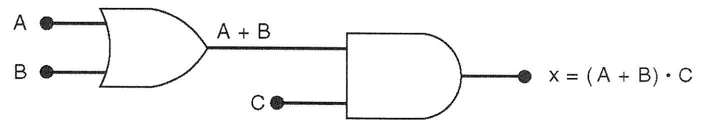


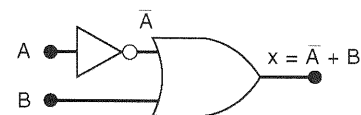
Fig. 3-13 Circuito lógico cuja expressão requer parênteses.

A fim de dar mais um exemplo, considere o circuito da Fig. 3-13. A expressão para a saída da porta OR é simplesmente $A + B$. Esta saída serve como entrada de uma porta AND juntamente com uma outra entrada *C*. Portanto, podemos expressar a saída da porta AND como $x = (A + B) \cdot C$. Observe o uso de parênteses para indicar que *A* OR *B* é realizada primeiro, isto é, antes que se faça um AND desta soma OR com *C*. Sem os parênteses, poderíamos interpretar a expressão de forma *incorreta*, uma vez que $A + B \cdot C$ significa *A* OR com o produto $B \cdot C$.

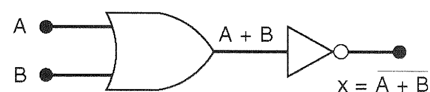
Circuitos Contendo INVERSORes

Sempre que um INVERSOR é apresentado em um diagrama de circuitos lógicos, a expressão para a sua saída é simplesmente igual à expressão da entrada com um barra sobre ela. A Fig. 3-14 mostra dois exemplos usando INVERSORes. Na Fig. 3-14(a), a entrada é conectada a um inversor, e a saída do mesmo é igual a $\overline{A}$. A saída do INVERSOR é conectada a uma porta OR juntamente com *B*, de modo que a saída da porta OR é igual a $\overline{A} + B$. Observe que a barra está apenas sobre o *A*, indicando que *A* é primeiramente invertido e depois é feita uma operação OR com *B*.

Na Fig. 3-14(b), a saída da porta OR é igual a $A + B$, e esta é conectada a um INVERSOR. A saída do INVERSOR é portanto igual a $(\overline{A + B})$, uma vez que ele inverte a expressão de entrada *completa*. Observe que a barra cobre a expressão $(A + B)$ inteira. Isto é importante porque, como será mostrado mais adiante, as expressões $(\overline{A + B})$ e $(\overline{A} + \overline{B})$ *não* são equivalentes. A expressão $(\overline{A + B})$ significa que realizamos a operação *A* OR *B* e que depois o resultado desta operação é invertido, enquanto a expressão $(\overline{A} + \overline{B})$ indica que *A* é invertido, *B* é invertido e somente depois é feita uma operação OR com estes resultados.



(a)



(b)

Fig. 3-14 Circuitos que usam INVERSORes.

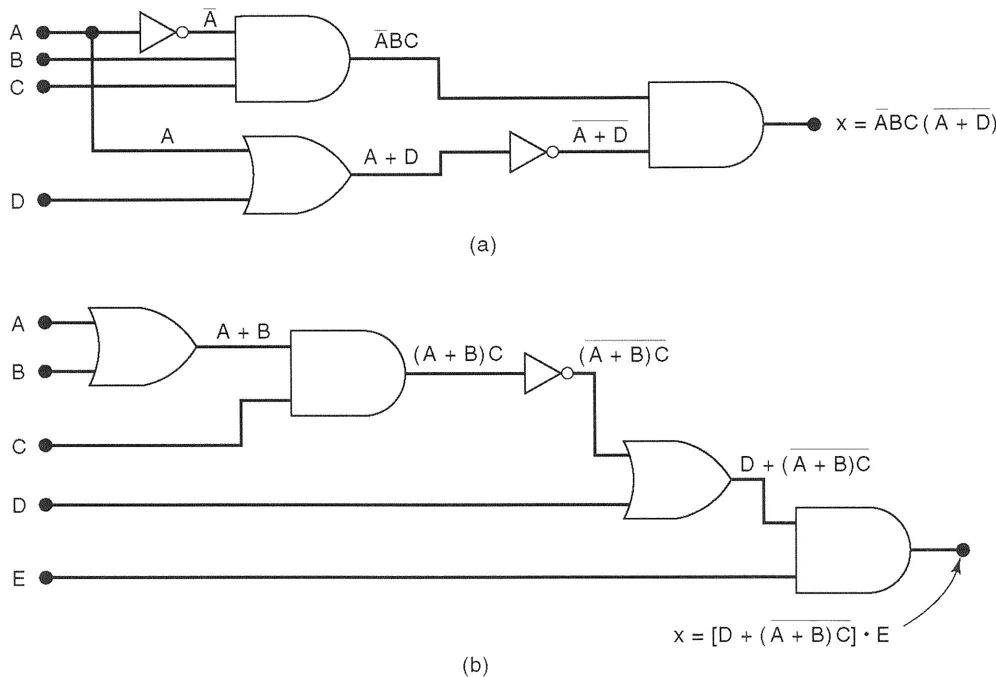


Fig. 3-15 Mais exemplos.

A Fig. 3-15 mostra mais dois exemplos que devem ser estudados com cuidado. Observe especialmente o uso de *dois* conjuntos separados de parênteses na Fig. 3-15(b). Observe também que na Fig. 3-15(a) a variável de entrada A está conectada como entrada em duas portas diferentes.

> **Questões de Revisão**
>
> **1.** Na Fig. 3-15(a), troque cada uma das portas AND por uma porta OR e troque a porta OR por uma porta AND. Agora escreva a expressão para a saída x.

3-7 DETERMINANDO O VALOR DA SAÍDA DE CIRCUITOS LÓGICOS

Uma vez obtida a expressão booleana para a saída do circuito, o nível lógico da saída pode ser determinado para qualquer conjunto de níveis lógicos das entradas. Por exemplo, suponha que desejamos saber o nível lógico da saída x para o circuito mostrado na Fig. 3-15(a), para o caso em que $A = 0$, $B = 1$, $C = 1$ e $D = 1$. Como na álgebra ordinária, o valor de x pode ser encontrado substituindo-se os valores das variáveis na expressão e fazendo as operações como se segue:

$$\begin{aligned} x &= \overline{A}BC(\overline{A + D}) \\ &= \overline{0} \cdot 1 \cdot 1 \cdot (\overline{0 + 1}) \\ &= 1 \cdot 1 \cdot 1 \cdot (\overline{0 + 1}) \\ &= 1 \cdot 1 \cdot 1 \cdot (\overline{1}) \\ &= 1 \cdot 1 \cdot 1 \cdot 0 \\ &= 0 \end{aligned}$$

Como um outro exemplo, vamos avaliar a expressão para a saída do circuito da Fig. 3-15(b), para o caso em que $A = 0$, $B = 0$, $C = 1$, $D = 1$ e $E = 1$.

$$\begin{aligned} x &= [D + \overline{(A + B)C}] \cdot E \\ &= [1 + \overline{(0 + 0) \cdot 1}] \cdot 1 \\ &= [1 + \overline{0 \cdot 1}] \cdot 1 \\ &= [1 + \overline{0}] \cdot 1 \\ &= [1 + 1] \cdot 1 \\ &= 1 \cdot 1 \\ &= 1 \end{aligned}$$

De um modo geral, as seguintes regras devem ser obedecidas quando avaliamos expressões booleanas:

1. Primeiro, faça todas as inversões de termos simples, isto é, $\overline{0} = 1$ ou $\overline{1} = 0$.
2. A seguir, faça todas as operações que estão dentro dos parênteses.
3. Faça a operação AND antes da operação OR, a não ser que os parênteses indiquem o contrário.
4. Se a expressão tiver uma barra sobreposta, faça as operações da expressão primeiro e depois inverta o resultado.

Para praticar, determine os níveis lógicos das saídas dos circuitos da Fig. 3-15 para o caso em que todas as entradas são iguais a 1. As respostas são $x = 0$ e $x = 1$, respectivamente.

Determinando o Nível da Saída a Partir de um Diagrama

O nível lógico da saída para um dado conjunto de níveis lógicos das entradas também pode ser determinado diretamente do diagrama do circuito, *sem* utilizar a expressão booleana. Esta técnica é freqüentemente usada por técnicos durante testes ou reparos de circuitos digitais, uma vez que ela mostra qual deveria ser a saída de cada porta, bem como qual deveria ser a saída final do sistema. Por exemplo, o circuito da Fig. 3-15(a) foi redesenhado na Fig. 3-16 com níveis de entrada iguais a $A = 0$, $B = 1$, $C = 1$, $D = 1$. O procedimento é o seguinte: a partir das entradas, devemos determinar

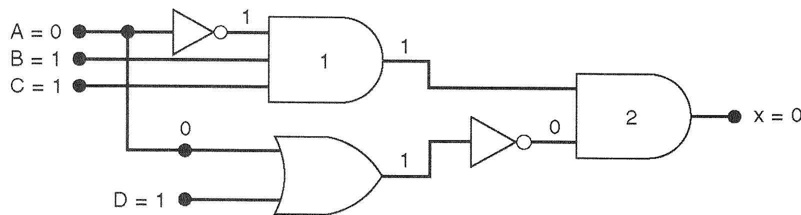


Fig. 3-16 Determinando o nível lógico de saída a partir do diagrama do circuito.

para cada INVERSOR, ou porta, o valor de sua saída até que o valor da saída final do sistema seja encontrado.

Na Fig. 3-16, a porta AND número 1 tem *todas* as suas entradas em nível 1 porque o INVERSOR troca $A = 0$ para 1. Esta condição produz um nível lógico 1 na saída da porta AND, uma vez que $1 \cdot 1 \cdot 1 = 1$. A porta OR tem como entradas os níveis 0 e 1, o que produz um nível 1 na saída, uma vez que $1 + 0 = 1$. Este nível 1 é invertido para nível 0, e este, por sua vez, é aplicado como entrada da porta AND número 2, juntamente com a saída da porta AND número 1. Os níveis 0 e 1 nas entradas da porta AND número 2 vão gerar na saída um nível lógico 0 porque $0 \cdot 1 = 0$.

EXEMPLO 3-6

Determine a saída do circuito da Fig 3-16 para o caso em que todas as entradas estão em BAIXO.

Solução

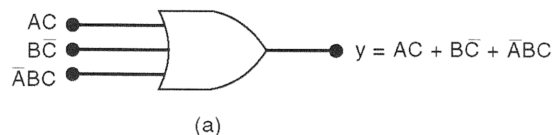
Com $A = B = C = D = 0$, a saída da porta AND 1 estará no nível BAIXO. Este é colocado na entrada da porta AND 2, o que automaticamente gera um nível BAIXO na saída, independentemente dos níveis lógicos em outros pontos do circuito. Este exemplo mostra que nem sempre é necessário determinar os níveis lógicos em todos os pontos do circuito para determinar o nível lógico de sua saída.

Questões de Revisão

1. Use a expressão para x para determinar a saída do circuito da Fig. 3-15(a), para as seguintes condições de entrada: $A = 0$, $B = 1$, $C = 1$ e $D = 0$.
2. Use a expressão para x para determinar a saída do circuito da Fig. 3-15(b), para as seguintes condições de entrada: $A = B = E = 1$ e $C = D = 0$.
3. Determine as respostas das questões 1 e 2, encontrando os níveis lógicos presentes em cada entrada e saída das portas lógicas, como foi feito na Fig. 3-16.

3-8 IMPLEMENTANDO CIRCUITOS A PARTIR DE EXPRESSÕES BOOLEANAS

Se a operação de um circuito lógico é definida por meio de uma expressão booleana, então o diagrama do circuito lógico pode ser implementado diretamente desta expressão. Por exemplo, se necessitamos de um circuito que é defini-



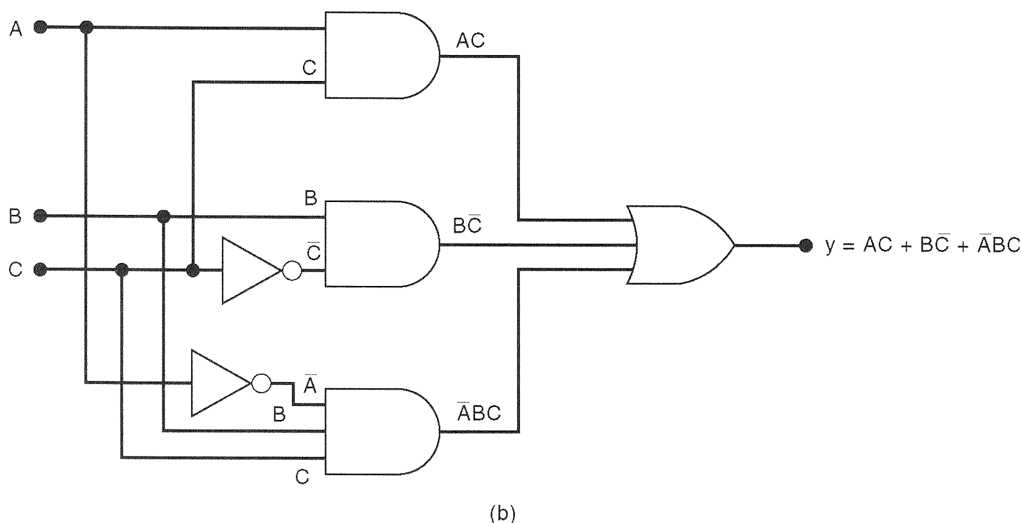


Fig. 3-17 Construindo um circuito lógico a partir de uma expressão booleana.

do pela expressão $x = A \cdot B \cdot C$, percebemos imediatamente que tudo de que precisamos é uma porta AND de três entradas. Se precisamos de um circuito definido pela expressão $x = A + \overline{B}$, poderíamos usar uma porta OR de duas entradas com um INVERSOR em uma de suas entradas. Esse mesmo raciocínio usado para esses casos simples pode ser estendido para circuitos mais complexos.

Suponha que desejamos implementar um circuito cuja saída pode ser definida pela expressão $y = AC + B\overline{C} + \overline{A}BC$. Esta expressão booleana possui três termos (AC, $B\overline{C}$, $\overline{A}BC$) sobre os quais é feita uma operação OR. Isto nos diz que necessitamos de uma porta OR de três entradas que são iguais a AC, $B\overline{C}$ e $\overline{A}BC$, respectivamente. Isto é mostrado na Fig. 3-17(a), onde uma porta OR de três entradas está desenhada com suas entradas AC, $B\overline{C}$ e $\overline{A}BC$.

Cada entrada da porta OR é um termo que expressa uma operação AND, o que significa que portas AND com entradas apropriadas devem ser usadas para gerar cada um desses termos. Isto é mostrado na Fig. 3-17(b) que é o diagrama do circuito final. Observe o uso de INVERSORes para produzir os termos $\overline{A}$ e $\overline{C}$ necessários à expressão.

Essa abordagem é bastante geral e pode ser sempre seguida, embora vamos ver mais tarde que existem outras técnicas melhores e mais eficientes que podem ser empregadas. Por enquanto, esse método direto de implementar circuitos lógicos deve ser utilizado para diminuir o número de coisas novas que devem ser aprendidas.

EXEMPLO 3-7

Desenhe o circuito que implementa a expressão $x = AB + \overline{B}C$.

Solução

Esta expressão indica que os termos AB e $\overline{B}C$ são entradas de uma porta OR, e cada um destes termos pode ser gerado por uma porta AND. O resultado é mostrado na Fig. 3-18.

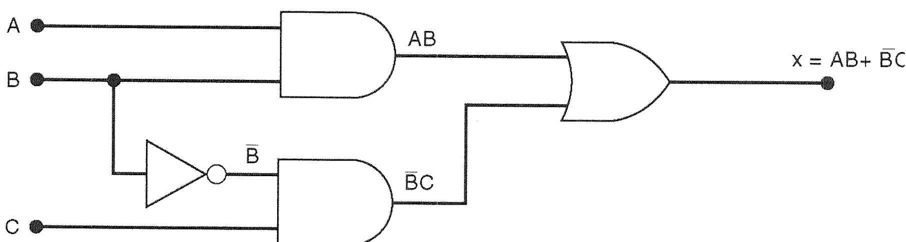


Fig. 3-18 Exemplo 3-7.

Questões de Revisão

1. Desenhe o circuito que implementa a expressão $x = \overline{A}BC(\overline{A+D})$, usando portas lógicas com no máximo três entradas.
2. Desenhe o circuito para a expressão $y = AC + B\overline{C} + \overline{A}BC$.
3. Desenhe o circuito para $x = [D + (\overline{A+B})C] \cdot E$.

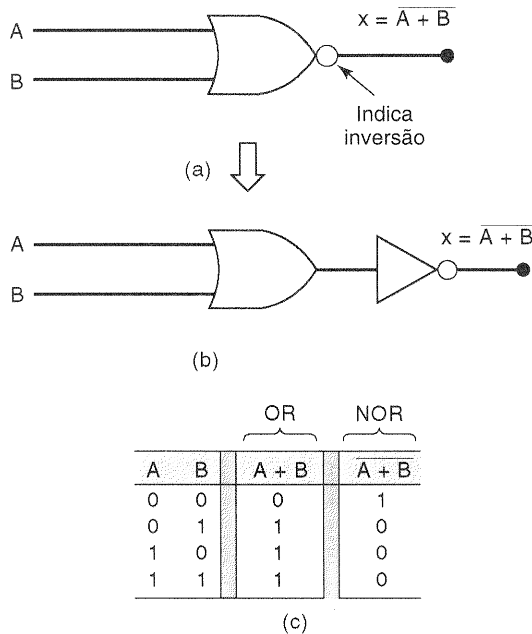
3-9 PORTAS NOR E PORTAS NAND

Existem dois outros tipos de portas lógicas, portas NOR e portas NAND, que são amplamente utilizadas em circuitos digitais. Estas portas, na verdade, combinam as operações básicas AND, OR e NOT. Este fato faz com que seja relativamente simples descrever o seu funcionamento utilizando as operações booleanas aprendidas anteriormente.

Porta NOR

O símbolo para uma **porta NOR** de duas entradas pode ser visto na Fig. 3-19(a). Este símbolo é igual ao símbolo de uma porta OR, exceto pelo pequeno círculo que possui em sua saída. Este pequeno círculo representa a operação de inversão. Então, podemos dizer que uma porta NOR opera do mesmo modo que uma porta OR seguida de um INVERSOR, de modo que os circuitos mostrados na Fig. 3-19(a) e (b) são equivalentes e a expressão booleana para a saída de uma porta NOR é dada por $x = \overline{A + B}$.

A tabela-verdade, que pode ser vista na Fig. 3-19(c), mostra que a saída de uma porta NOR é exatamente o inverso da saída para uma porta OR, para todas as condições de entrada. Enquanto a saída de uma porta OR vai para o nível ALTO sempre que qualquer uma das entradas está em ALTO, a porta NOR vai para nível BAIXO sempre que qualquer uma das entradas está em ALTO. Este mesmo raciocínio pode ser estendido para portas NOR com mais de duas entradas.



A	B	A + B	$\overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Fig. 3-19 (a) Símbolo para porta NOR; (b) circuito equivalente; (c) tabela-verdade.

EXEMPLO 3-8

Determine a forma de onda da saída de uma porta NOR para as formas de onda mostradas na Fig. 3-20.

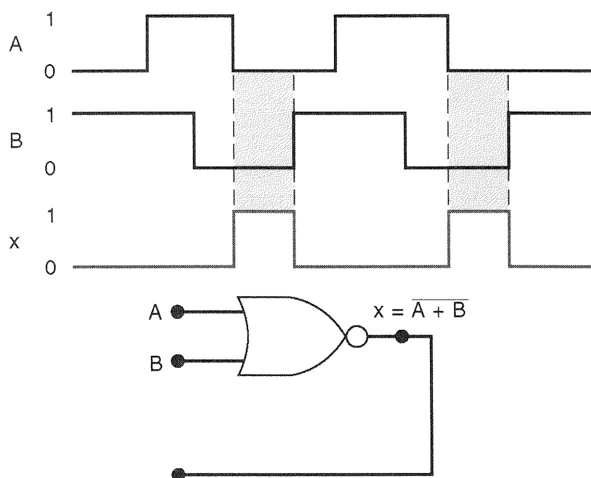


Fig. 3-20 Exemplo 3-8.

Solução

Existem diversas maneiras de determinar a forma de onda da saída de uma porta NOR. A primeira é encontrar a forma de onda da saída de uma porta OR e depois invertê-la, isto é, trocar todos os 1s por 0s e vice-versa. Uma outra utiliza o fato de que a saída de uma porta NOR estará em ALTO *apenas* quando todas as entradas estiverem em BAIXO. Então você pode examinar as formas de onda das entradas e encontrar os intervalos de tempo em que todas as entradas estão em BAIXO e fazer com que a saída esteja em ALTO nestes intervalos. A saída da porta NOR estará em BAIXO para todos os outros intervalos de tempo. A forma de onda resultante para a saída é mostrada na figura.

EXEMPLO 3-9

Determine a expressão booleana para uma porta NOR de três entradas seguida de um INVERSOR.

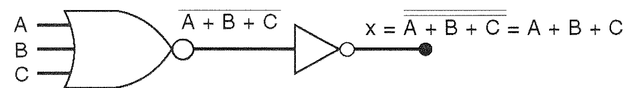


Fig. 3-21 Exemplo 3-9.

Solução

Observe a Fig. 3-21, onde o diagrama do circuito pode ser visto. A expressão para a saída da porta NOR é dada por $(\overline{A + B + C})$. Esta saída está conectada na entrada de um INVERSOR para produzir:

$$x = \overline{(\overline{A + B + C})}$$

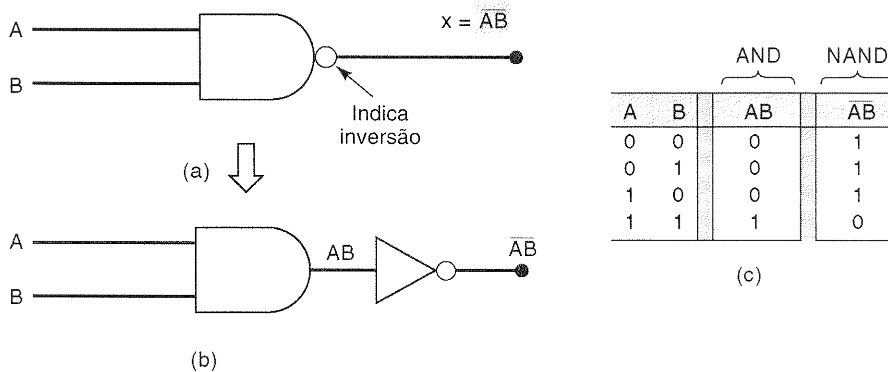
A presença de dois sinais de inversão indica que a expressão $(A + B + C)$ foi invertida e depois invertida mais uma vez. Deve estar claro que o resultado destas operações simplesmente não altera a expressão original $(A + B + C)$. Isto é,

$$x = \overline{(\overline{A + B + C})} = A + B + C$$

Sempre que duas barras de inversão estiverem sobre uma mesma variável ou expressão, elas se cancelam, como no exemplo anterior. Entretanto, em casos como $\overline{\overline{A} + \overline{B}}$, as barras de inversão não se cancelam. Isto acontece porque as barras de inversão menores invertem as variáveis simples A e B, respectivamente, enquanto as barras mais largas invertem toda a expressão $(\overline{A} + \overline{B})$. Então $\overline{\overline{A} + \overline{B}} \neq A + B$. De modo semelhante, $\overline{\overline{A} + \overline{B}} \neq AB$.

Porta NAND

O símbolo para uma **porta NAND** de duas entradas pode ser visto na Fig. 3-22(a). Este símbolo é igual ao símbolo da porta AND, exceto pelo pequeno círculo em sua saída. Uma vez mais, este pequeno círculo representa uma operação de inversão. Então, podemos dizer que uma porta NAND funciona como uma porta AND seguida de um INVERSOR, e que portanto os circuitos das Fig. 3-22(a) e (b) são equivalentes e que a expressão booleana para a saída de uma porta NAND é $x = \overline{AB}$.



		AND	NAND
A	B	AB	$\overline{AB}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

(c)

Fig. 3-22 (a) Símbolo para porta NAND; (b) circuito equivalente; (c) tabela-verdade.

A tabela-verdade vista na Fig. 3-22(c) mostra que a saída de uma porta NAND é exatamente o inverso da saída de uma porta AND para todas as condições possíveis de entrada. A saída de uma porta AND vai para ALTO quando todas as entradas estão em ALTO, enquanto a saída de uma porta NAND vai para BAIXO somente quando todas as entradas estão em ALTO. Portas NAND com mais de duas entradas também apresentam essa mesma característica.

EXEMPLO 3-10

Determine a forma de onda da saída de uma porta NAND cujas formas de onda das entradas estão mostradas na Fig. 3-23.

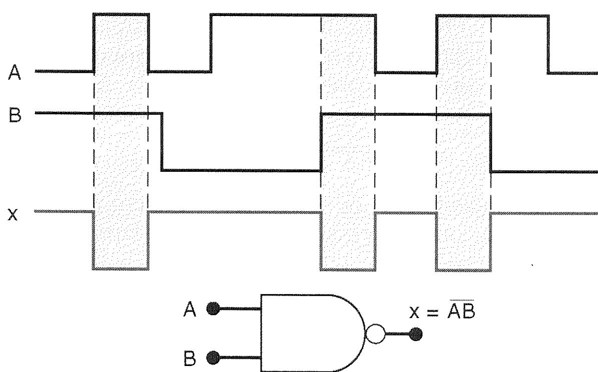


Fig. 3-23 Exemplo 3-10.

Solução

A forma de onda da saída pode ser determinada de várias maneiras. Uma delas é desenhar a forma de onda da saída para o caso de uma porta AND e depois inverter o resultado. Uma outra utiliza o fato de que a saída da porta NAND estará em BAIXO apenas quando todas as entradas estiverem em ALTO. Então, você pode encontrar os intervalos de tempo durante os quais todas as entradas estão em ALTO e fazer com que a saída esteja em BAIXO nesses intervalos. A saída estará em ALTO em todos os outros intervalos.

EXEMPLO 3-11

Implemente um circuito lógico cuja expressão é $x = \overline{AB \cdot (\overline{C + D})}$ usando apenas portas NAND e NOR.

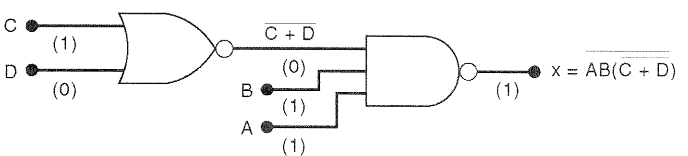


Fig. 3-24 Exemplos 3-11 e 3-12.

Solução

O termo $(\overline{C + D})$ é a expressão para a saída de uma porta NOR. Este termo, em conjunto com A e B, é utilizado como entrada de uma operação AND cujo resultado final é invertido. Isto, obviamente, resulta em uma operação NAND. Assim, o circuito implementado é aquele que pode ser visto na Fig. 3-24. Observe que a porta NAND primeiro realiza uma operação AND sobre os termos A, B e $(\overline{C+D})$ e depois inverte o resultado *inteiro*.

EXEMPLO 3-12

Determine o nível lógico da saída na Fig. 3-24 quando $A = B = C = 1$ e $D = 0$.

Solução

Podemos solucionar este problema de dois modos: no primeiro modo usamos a expressão booleana para x.

$$
\begin{aligned}
x &= \overline{AB\overline{(C + D)}} \\
&= \overline{1 \cdot 1 \cdot \overline{(1 + 0)}} \\
&= \overline{1 \cdot 1 \cdot \overline{(1)}} \\
&= \overline{1 \cdot 1 \cdot 0} \\
&= \overline{0} = 1
\end{aligned}
$$

No segundo método, escrevemos os níveis lógicos de entrada no diagrama do circuito (mostrados entre parênteses na Fig. 3-24) e a partir desses níveis achamos os níveis lógicos da saída de cada porta até encontrarmos o resultado final. A porta NOR possui como entradas 0 e 1, o que faz a saída ser igual a 0 (em uma porta OR a saída seria 1). A porta NAND então tem como entradas os níveis lógicos 0, 1 e 1, o que faz com que a saída seja igual a 1 (em uma porta AND a saída seria igual a 0).

Questões de Revisão

1. Qual é o único conjunto de condições de entrada que vai gerar um nível ALTO na saída de uma porta NOR de três entradas?
2. Determine o nível da saída do circuito da Fig 3-24 para o caso em que $A = B = 1$ e $C = D = 0$.
3. Troque a porta NOR da Fig. 3-24 por uma porta NAND e troque também a porta NAND por uma porta NOR. Qual é a nova expressão booleana para x?

3-10 TEOREMAS DA ÁLGEBRA BOOLEANA

Vimos como a álgebra booleana pode ser usada para nos ajudar a analisar um circuito lógico e expressar sua operação matematicamente. Continuaremos nosso estudo da álgebra booleana investigando seus vários teoremas (regras), chamados **teoremas booleanos,** que podem nos ajudar a simplificar expressões e circuitos lógicos. O primeiro grupo de teoremas é mostrado na Fig. 3-25. Em cada um deles, x é uma variável lógica que pode ser igual a 0 ou 1. Cada teorema está acompanhado por um circuito lógico que demonstra sua validade.

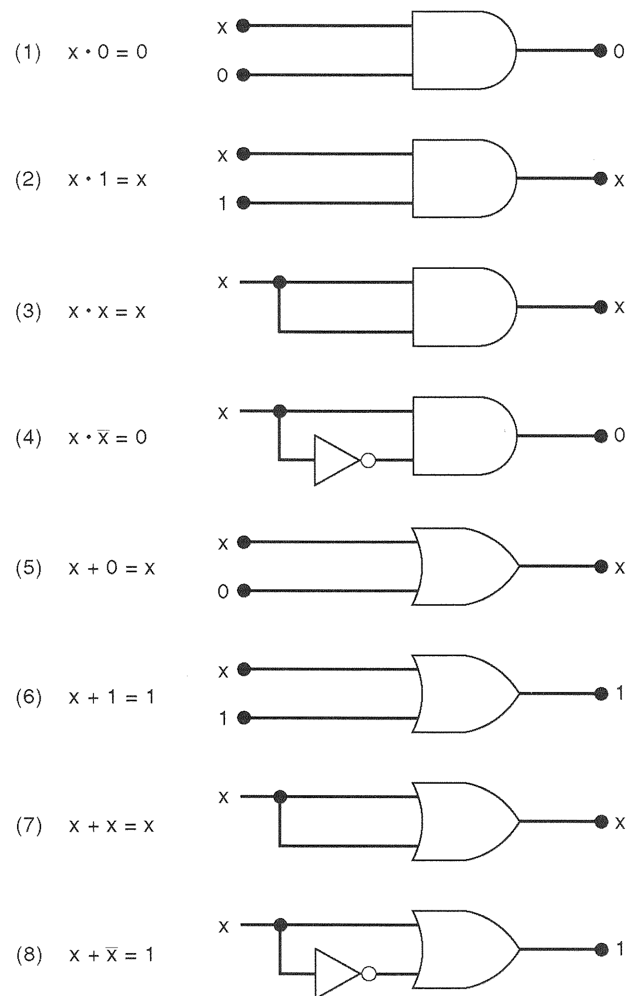


Fig. 3-25 Teoremas de uma variável.

O teorema (1) mostra que o resultado de uma operação AND que tem como entradas uma variável qualquer x e 0 deve ser igual a 0. Isto é fácil de lembrar porque a operação AND é como a multiplicação ordinária, onde sabemos que o resultado de multiplicar qualquer coisa por 0 é 0. Sabemos também que a saída de uma porta AND será 0 sempre que qualquer uma das entradas for 0, independentemente do nível lógico da outra entrada.

O teorema (2) é também óbvio, se fizermos mais uma vez a comparação da multiplicação ordinária com a operação AND.

O teorema (3) pode ser provado verificando o resultado para cada valor possível de entrada. Se $x = 0$, então $0 \cdot 0 = 0$; se $x = 1$, então $1 \cdot 1 = 1$. Portanto, $x \cdot x = x$.

O teorema (4) pode ser provado do mesmo modo. Entretanto, podemos raciocinar que em qualquer instante ou x ou seu inverso $\overline{x}$ deve ser igual a 0 e, então, uma operação AND de x com seu inverso será sempre igual a 0.

O teorema (5) é direto, uma vez que 0 *adicionado* a qualquer valor não altera esse valor, seja na adição ordinária ou na operação OR.

O teorema (6) afirma que o resultado de uma operação OR que possui como entradas uma variável qualquer x e 1 será sempre igual a 1. Podemos fazer a verificação deste teorema para os dois valores possíveis de x: $0 + 1 = 1$ e $1 + 1 = 1$. De modo equivalente, podemos lembrar que a saída de uma porta OR de duas entradas será igual a 1 quando *qualquer* uma das entradas for igual a 1, não importando o valor da outra entrada.

O teorema (7) pode ser verificado para ambos os valores de x: $0 + 0 = 0$ e $1 + 1 = 1$.

O teorema (8) pode ser provado de modo similar, ou podemos raciocinar que em qualquer instante x ou seu inverso $\overline{x}$ estará em nível lógico 1, então sempre teremos a operação OR de 0 e 1, cujo resultado será sempre 1.

Antes de introduzirmos mais teoremas, devemos enfatizar que quando os teoremas de (1) a (8) são aplicados, a variável x pode, na verdade, representar uma expressão que contenha mais de uma variável. Por exemplo, se tivermos a expressão $A\overline{B}(\overline{A\overline{B}})$, podemos aplicar o teorema (4) se fizermos $x = A\overline{B}$. Então podemos dizer que $A\overline{B}(\overline{A\overline{B}}) = 0$. Este mesmo raciocínio pode ser aplicado para o uso de qualquer um destes teoremas.

Teoremas com Mais de Uma Variável

Os teoremas apresentados a seguir envolvem o uso de mais de uma variável:

(9) $x + y = y + x$
(10) $x \cdot y = y \cdot x$
(11) $x + (y + z) = (x + y) + z = x + y + z$
(12) $x(yz) = (xy)z = xyz$
(13*a*) $x(y + z) = xy + xz$
(13*b*) $(w + x)(y + z) = wy + xy + wz + xz$
(14) $x + xy = x$
(15) $x + \overline{x}y = x + y$

Os teoremas (9) e (10) são conhecidos como *leis da comutatividade*. Estas leis determinam que a ordem na qual realizamos as operações AND e OR não é importante. O resultado é o mesmo.

Os teoremas (11) e (12) são conhecidos como *leis da associatividade*; elas afirmam que podemos agrupar as variáveis de expressões do tipo AND ou OR do modo que desejarmos.

O teorema (13) é a *lei da distributividade*, que afirma que uma expressão pode ser expandida multiplicando-se termo a termo, do mesmo modo que é feito na álgebra comum. Este teorema também afirma que podemos fatorar uma expressão. Caso tenhamos a soma de dois (ou mais) termos, cada um contendo uma variável comum, podemos fatorar essa variável como fazemos na álgebra comum. Por exemplo, na expressão $A\overline{B}C + \overline{A}\,\overline{B}\,\overline{C}$, podemos fatorar a variável $\overline{B}$:

$$A\overline{B}C + \overline{A}\overline{B}\overline{C} = \overline{B}(AC + \overline{A}\overline{C})$$

Como um outro exemplo, considere a expressão $ABC + ABD$. Neste caso, estes dois termos têm as variáveis A e B em comum, e portanto $A \cdot B$ pode ser fatorado, como vemos a seguir.

$$ABC + ABD = AB(C + D)$$

Os teoremas (9) a (13) são fáceis de lembrar porque são idênticos àqueles utilizados na álgebra comum. Os teoremas (14) e (15), por outro lado, não possuem correspondentes na álgebra comum. Cada um deles pode ser demonstrado substituindo x e y na expressão por todos os diferentes casos possíveis, conforme demonstrado para o teorema (14):

Caso 1. Para $x = 0$, $y = 0$,1

$$\begin{aligned} x + xy &= x \\ 0 + 0 \cdot 0 &= 0 \\ 0 &= 0 \end{aligned}$$

Caso 2. Para $x = 0$, $y = 1$,

$$\begin{aligned} x + xy &= x \\ 0 + 0 \cdot 1 &= 0 \\ 0 + 0 &= 0 \\ 0 &= 0 \end{aligned}$$

Caso 3. Para $x = 1$, $y = 0$

$$\begin{aligned} x + xy &= x \\ 1 + 1 \cdot 0 &= 1 \\ 1 + 0 &= 1 \\ 1 &= 1 \end{aligned}$$

Caso 4. Para $x = 1$, $y = 1$,

$$\begin{aligned} x + xy &= x \\ 1 + 1 \cdot 1 &= 1 \\ 1 + 1 &= 1 \\ 1 &= 1 \end{aligned}$$

O teorema (14) também pode ser demonstrado através de fatoração e do uso dos teoremas (6) e (2).

$$\begin{aligned} x + xy &= x(1 + y) \\ &= x \cdot 1 \quad \text{[usando o teorema (6)]} \\ &= x \quad \text{[usando o teorema (2)]} \end{aligned}$$

Todos esses teoremas da álgebra booleana podem ser úteis na simplificação de uma expressão lógica, isto é, na redução do número de termos da expressão. Quando isto é feito, a expressão simplificada dá origem a um circuito que é menos complexo do que aquele que a expressão original produziria. Uma boa parte do próximo capítulo será dedicada ao processo de simplificação de circuitos. Por enquanto, os exemplos seguintes servem para ilustrar como os teoremas booleanos podem ser aplicados.

EXEMPLO 3-13

Simplifique a expressão $y = A\overline{B}D + A\overline{B}\,\overline{D}$.

Solução

Fatore as variáveis comuns $A\overline{B}$ utilizando o teorema (13):

$$y = A\overline{B}(D + \overline{D})$$

Pelo teorema (8), o termo entre parênteses é igual a 1 e portanto,

$$\begin{aligned} y &= A\overline{B} \cdot 1 \\ &= A\overline{B} \quad \text{[usando o teorema (2)]} \end{aligned}$$

EXEMPLO 3-14

Simplifique $z = (\overline{A} + B)(A + B)$.

Solução

A expressão pode ser expandida multiplicando-se os termos [teorema (13)]:

$$z = \overline{A} \cdot A + \overline{A} \cdot B + B \cdot A + B \cdot B$$

Pelo teorema (4), $\overline{A} \cdot A = 0$. Além disso, $B \cdot B = B$ [teorema (3)]:

$$z = 0 + \overline{A} \cdot B + B \cdot A + B = \overline{A}B + AB + B$$

Fatorando a variável B [teorema (13)], temos

$$z = B(\overline{A} + A + 1)$$

Finalmente, utilizando os teoremas (2) e (6),

$$z = B$$

EXEMPLO 3-15

Simplifique $x = ACD + \overline{A}BCD$.

Solução

Fatorando as variáveis comuns CD, temos

$$x = CD(A + \overline{A}B)$$

Utilizando o teorema (15), podemos substituir $A + \overline{A}B$ por $A + B$, e então

$$x = CD(A + B)$$
$$= ACD + BCD$$

Questões de Revisão

1. Use os teoremas (13) e (14) para simplificar $y = A\overline{C} + AB\overline{C}$.
2. Use os teoremas (13) e (8) para simplificar $y = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$.

3-11 TEOREMAS DE DEMORGAN

Dois dos mais importantes teoremas da álgebra booleana são atribuídos a um grande matemático chamado DeMorgan. Os **teoremas de DeMorgan** são extremamente úteis para simplificar expressões nas quais o produto (AND) ou a soma (OR) das variáveis é invertido. Os dois teoremas são:

$$(16) \qquad \overline{(x + y)} = \overline{x} \cdot \overline{y}$$
$$(17) \qquad \overline{(x \cdot y)} = \overline{x} + \overline{y}$$

O teorema (16) diz que quando uma soma OR está invertida, esta é igual ao produto AND das variáveis invertidas. O teorema (17) diz que quando um produto AND de duas variáveis está invertido, este é igual a uma soma OR das variáveis invertidas. Cada um dos teoremas de DeMorgan pode ser prontamente demonstrado verificando-o para todas as combinações possíveis de valores para x e y. Esta demonstração é deixada para ser feita como exercício ao final do capítulo.

Apesar de esses teoremas terem sido enunciados em termos de variáveis simples x e y, eles são igualmente válidos para situações nas quais x e/ou y são expressões que contenham mais de uma variável. Por exemplo, a aplicação destes teoremas na expressão $\overline{(A\overline{B} + C)}$ pode ser vista a seguir:

$$\overline{(A\overline{B} + C)} = (\overline{A\overline{B}}) \cdot \overline{C}$$

Note que aqui tratamos $A\overline{B}$ como x e C como y. O resultado pode depois ser simplificado já que temos um produto $A\overline{B}$ que é invertido. Usando o teorema (17), a expressão se torna

$$\overline{A\overline{B}} \cdot \overline{C} = (\overline{A} + \overline{\overline{B}}) \cdot \overline{C}$$

Observe que podemos substituir $\overline{\overline{B}}$ por B, e então finalmente temos

$$(\overline{A} + B) \cdot \overline{C} = \overline{A}\,\overline{C} + B\overline{C}$$

Este resultado final possui sinais de inversão apenas em variáveis simples.

EXEMPLO 3-16

Simplifique a expressão $z = \overline{(\overline{A}+C) \cdot (B+\overline{D})}$ para uma outra que contenha apenas variáveis simples invertidas.

Solução

Utilizando o teorema (17), podemos reescrever a expressão anterior como

$$z = (\overline{\overline{A} + C}) + (\overline{B + \overline{D}})$$

Podemos pensar nesse procedimento como partir o sinal de inversão ao meio e trocar sinais AND (·) por sinais OR (+). Agora o termo $(\overline{\overline{A} + C})$ pode ser simplificado aplicando-se o teorema (16). Do mesmo modo, $(\overline{B+\overline{D}})$ pode ser simplificado como se segue:

$$z = (\overline{\overline{A} + C}) + (\overline{B + \overline{D}})$$
$$= (\overline{\overline{A}} \cdot \overline{C}) + \overline{B} \cdot \overline{\overline{D}}$$

Nesta simplificação, partimos o sinal de inversão ao meio e trocamos os sinais (+) por (·). A seguir, cancelamos as inversões duplas e temos finalmente

$$z = A\overline{C} + \overline{B}D$$

O Exemplo 3-16 mostra que, quando se utilizam os teoremas de DeMorgan para simplificar uma expressão, o que fazemos é partir o sinal de inversão em qualquer ponto na expressão e então mudar o sinal do operador que estiver neste ponto (+ é trocado por · e vice-versa). Este procedimento pode ser continuado até que a expressão seja reduzida a uma outra na qual apenas variáveis simples encontram-se invertidas. Outros dois exemplos podem ser vistos a seguir:

Exemplo 1

$$z = \overline{A + \overline{B} \cdot C}$$
$$= \overline{A} \cdot \overline{(\overline{B} \cdot C)}$$
$$= \overline{A} \cdot (\overline{\overline{B}} + \overline{C})$$
$$= \overline{A} \cdot (B + \overline{C})$$

Exemplo 2

$$\omega = \overline{(A + BC) \cdot (D + EF)}$$
$$= \overline{(A + BC)} + \overline{(D + EF)}$$
$$= (\overline{A} \cdot \overline{BC}) + (\overline{D} \cdot \overline{EF})$$
$$= [\overline{A} \cdot (\overline{B} + \overline{C})] + [\overline{D} \cdot (\overline{E} + \overline{F})]$$
$$= \overline{A}\,\overline{B} + \overline{A}\,\overline{C} + \overline{D}\,\overline{E} + \overline{D}\,\overline{F}$$

Os teoremas de DeMorgan podem ser facilmente estendidos para mais do que duas variáveis. Por exemplo, pode-se provar que:

$$\overline{x + y + z} = \overline{x} \cdot \overline{y} \cdot \overline{z}$$
$$\overline{x \cdot y \cdot z} = \overline{x} + \overline{y} + \overline{z}$$

Aqui podemos ver que o grande sinal de inversão foi partido em *dois* pontos e, nesses pontos, o sinal do operador foi trocado por seu oposto. Esse raciocínio pode ser estendido para um número qualquer de variáveis. Mais uma vez, observe que as variáveis podem ser expressões em lugar de variáveis simples. Veja um outro exemplo:

$$x = \overline{\overline{AB} \cdot \overline{CD} \cdot \overline{EF}}$$
$$= \overline{\overline{AB}} + \overline{\overline{CD}} + \overline{\overline{EF}}$$
$$= AB + CD + EF$$

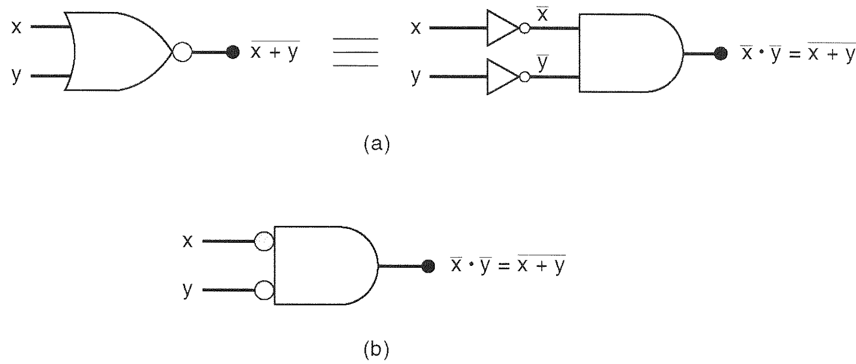


Fig. 3-26 (a) Circuitos equivalentes obtidos pela aplicação do teorema (16); (b) símbolo alternativo para a função NOR.

Implicações dos Teoremas de DeMorgan

Vamos examinar os teoremas (16) e (17) do ponto de vista de circuitos lógicos. Primeiro considere o teorema (16):

$$\overline{x + y} = \bar{x} \cdot \bar{y}$$

O lado esquerdo da equação pode ser visto como a saída de uma porta NOR cujas entradas são x e y. O lado direito da equação, por outro lado, pode ser visto como a saída de uma porta AND cujas entradas são as variáveis x e y invertidas. Estas duas representações são equivalentes e estão ilustradas na Fig. 3-26(a). Isto significa que uma porta AND com inversores em cada uma de suas entradas é equivalente a uma porta NOR. Na verdade, ambas as representações são usadas para representar a função NOR. Quando a porta AND com entradas invertidas é usada para representar a função NOR, esta é geralmente desenhada como mostrado na Fig. 3-26(b), onde os pequenos círculos nas entradas representam a operação de inversão.

Agora considere o teorema (17):

$$\overline{x \cdot y} = \bar{x} + \bar{y}$$

O lado esquerdo da equação pode ser implementado através de uma porta NAND com entradas x e y. O lado direito pode ser implementado por uma porta OR que tenha como entradas x e y invertidas. Estas duas representações equivalentes são mostradas na Fig.3-27(a). Uma porta OR com inversores em cada uma de suas entradas é equivalente a uma porta NAND. Na verdade, ambas as representações são usadas para representar a função NAND. Quando a porta OR com entradas invertidas é usada para representar a função NAND, esta é freqüentemente desenhada como mostra a Fig. 3-27(b), onde os círculos mais uma vez representam inversão.

EXEMPLO 3-17

Determine a expressão lógica para a saída do circuito da Fig. 3-28 e simplifique-a usando os teoremas de DeMorgan.

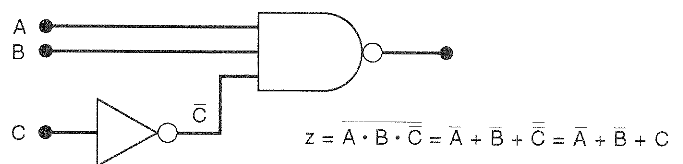


Fig. 3-28 Exemplo 3-17.

Solução

A expressão para z é $z = \overline{AB\bar{C}}$. Aplique o teorema de DeMorgan para partir o sinal de inversão como é mostrado a seguir:

$$z = \bar{A} + \bar{B} + \bar{\bar{C}}$$

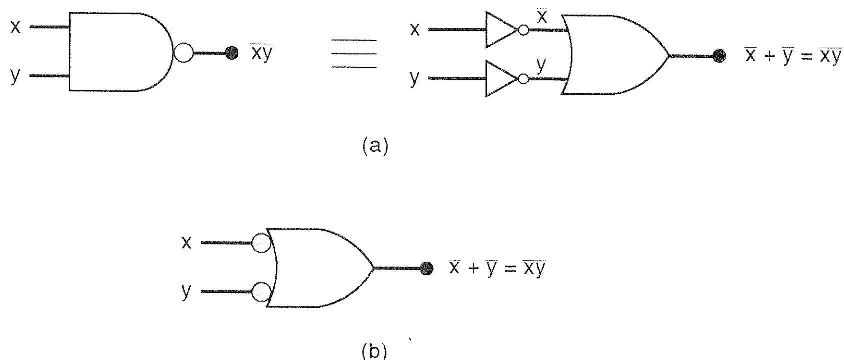


Fig. 3-27 (a) Circuitos equivalentes obtidos pela aplicação do teorema (17); (b) símbolo alternativo para a função NAND.

Cancele a dupla inversão sobre C para obter

$$z = \overline{A} + \overline{B} + C$$

Questões de Revisão

1. Use o teorema de DeMorgan para converter a expressão $z = \overline{(A+B)\cdot\overline{C}}$ para uma outra que possua apenas inversões em variáveis simples.
2. Repita a questão 1 para a expressão $y = \overline{R\overline{S}T+\overline{Q}}$.
3. Implemente um circuito cuja expressão para a saída é $z = \overline{\overline{A}\,\overline{B}C}$ usando apenas uma porta NOR e um INVERSOR.
4. Use os teoremas de DeMorgan para converter $y = \overline{\overline{A+\overline{B}+\overline{C}D}}$ para uma outra expressão que contenha apenas inversões em variáveis simples.

3-12 UNIVERSALIDADE DAS PORTAS NAND E NOR

Todas as expressões booleanas consistem em várias combinações das operações básicas OR, AND e NOT. Assim, qualquer expressão pode ser implementada usando combinações das portas AND, OR e INVERSORes. É possível, entretanto, implementar qualquer expressão lógica usando-se *apenas* portas NAND. Isto acontece porque portas NAND, em combinações apropriadas, podem ser usadas para representar cada uma das operações lógicas OR, AND e NOT. Isto pode ser visto na Fig. 3-29.

Em primeiro lugar, na Fig. 3-29(a), temos uma porta NAND de duas entradas onde estas se encontram propositadamente conectadas a uma mesma variável A. Nessa configuração, a porta NAND simplesmente age como um simples INVERSOR, uma vez que sua saída $x = \overline{A\cdot A} = \overline{A}$.

Na Fig. 3-29(b) temos duas portas NAND conectadas de tal modo que a operação AND seja realizada. A porta NAND número 2 é usada como um INVERSOR para que a expressão $\overline{AB}$ seja transformada em $\overline{\overline{AB}} = AB$, que é a função AND desejada.

A operação OR pode ser implementada usando portas NAND como está mostrado na Fig. 3-29(c). Neste caso, as portas NAND número 1 e 2 são usadas como INVERSORes para inverter as entradas, de modo que o resultado final seja $x = \overline{\overline{A}\cdot\overline{B}}$, que pode ser simplificado para $x = A + B$ através do teorema de DeMorgan.

De modo similar, pode ser mostrado que portas NOR podem ser combinadas para implementar qualquer uma das operações booleanas. Isto é ilustrado na Fig. 3-30. O item (a) mostra que uma porta NOR com as entradas conectadas juntas comporta-se como um INVERSOR, uma vez que sua saída é $x = \overline{A+A} = \overline{A}$.

Na Fig. 3-30(b), duas portas NOR são combinadas de modo a implementar a operação OR. A porta NOR número 2 é usada como um INVERSOR para modificar a expressão $\overline{A+B}$ em $\overline{\overline{A+B}} = A + B$, que é a operação OR desejada.

A operação AND pode ser implementada com portas NOR como pode ser visto na Fig. 3-30(c). Neste caso, as portas NOR 1 e 2 são usadas como INVERSORes para inverter as entradas, de modo que a saída x seja igual a $x = \overline{\overline{A}+\overline{B}}$, que pode ser simplificado para $x = A \cdot B$, pelo uso do teorema de DeMorgan.

Uma vez que qualquer uma das operações booleanas pode ser implementada usando apenas portas NAND, qualquer circuito lógico pode ser construído usando apenas portas NAND. O mesmo é válido para portas NOR. Essa característica das portas NAND e NOR pode ser bastante útil no projeto de circutos lógicos, como mostra o exemplo a seguir.

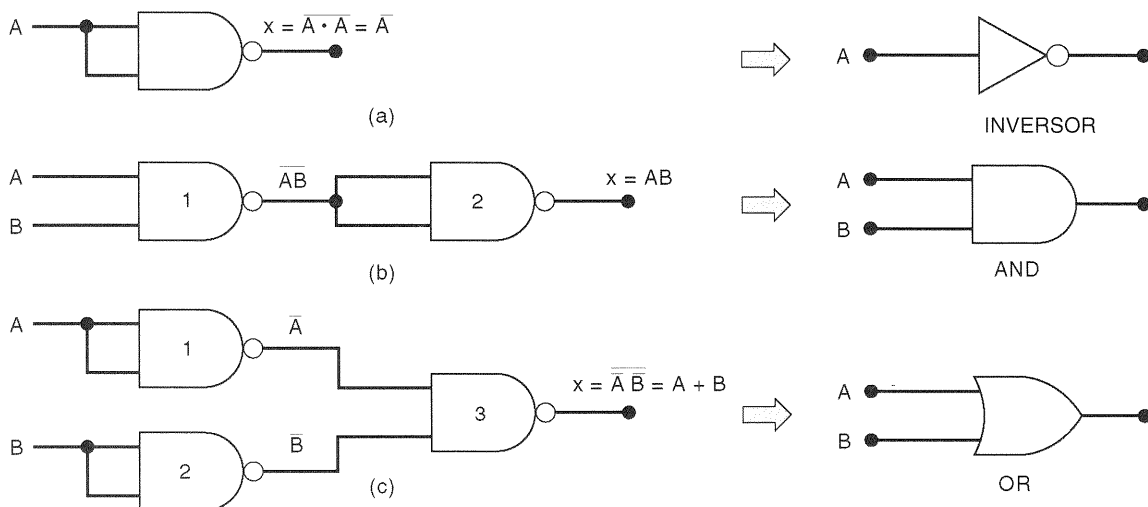


Fig 3-29 Portas NAND podem ser usadas para implementar qualquer função booleana.

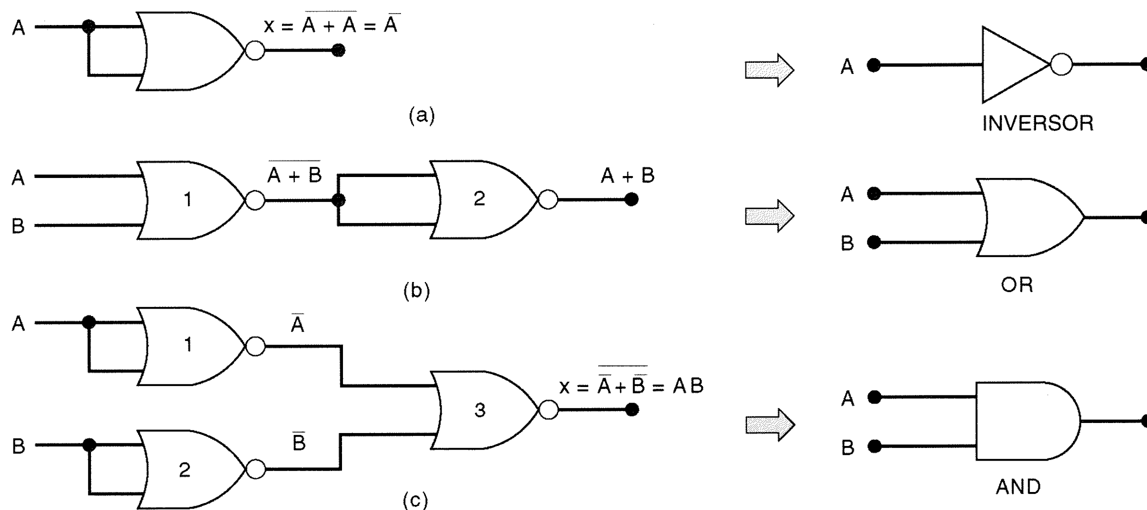


Fig. 3-30 Portas NOR podem ser usadas para implementar qualquer função booleana.

EXEMPLO 3-18

Em um determinado processo de fabricação, uma esteira de transporte deve ser desligada sempre que determinadas condições ocorrerem. Estas condições são monitoradas e são representadas pelo estado de quatro sinais lógicos como se segue: o sinal *A* deve estar em nível ALTO sempre que a esteira de transporte estiver muito rápida; o sinal *B* deve estar em nível ALTO sempre que o recipiente localizado no final da esteira estiver cheio; o sinal *C* deve estar em nível ALTO sempre que a tensão na esteira estiver muito alta; o sinal *D* deve estar em nível ALTO sempre que o comando manual estiver desabilitado.

Um circuito lógico é necessário para gerar um sinal *x* que deve estar em ALTO sempre que as condições *A* e *B* existirem simultaneamente, ou sempre que as condições *C* e *D* existirem simultaneamente. Obviamente, a expressão lógica para *x* deve ser igual a $x = AB + CD$. O circuito deve ser implementado com um número mínimo de CIs. Os circuitos integrados TTL mostrados na Fig. 3-31 estão disponíveis. Cada CI é *quádruplo*, o que significa que ele contém *quatro* portas idênticas em um chip.

Solução

O método mais direto para se implementar a expressão dada usa duas portas AND e uma porta OR, como pode ser visto na Fig. 3-32(a). Esta implementação utiliza duas portas do CI 74LS08 e uma única porta do CI 74LS32. Os números entre parênteses, em cada entrada e saída, são os números dos pinos dos respectivos CIs. Estes números são sempre mostrados em qualquer diagrama de circuito lógico. Para os nossos propósitos, a maioria dos diagramas lógicos não mostrará o número dos pinos, a não ser que eles sejam necessários para descrever a operação do circuito.

Uma outra implementação pode ser obtida a partir do circuito da Fig. 3-32(a), se trocarmos cada uma das portas AND e OR pelas suas implementações com portas NAND equivalentes. O resultado desta operação é mostrado na Fig. 3-32(b).

À primeira vista, esse novo circuito parece necessitar de sete portas NAND. Entretanto, as portas NAND de número 3 e 5 estão conectadas como INVERSORES em série e podem ser eliminadas do circuito, uma vez que realizam uma dupla inversão da saída da porta NAND número 1. Do mesmo modo, as portas NAND 4 e 6 também podem ser eliminadas. O circuito final, após a eliminação dos INVERSORes duplos, está desenhado na Fig. 3-32(c).

Esse circuito é mais eficiente do que o da Fig. 3-32(a) porque utiliza três portas NAND de duas entradas que podem ser implementadas por um CI, o 74LS00.

Questões de Revisão

1. Quantas maneiras diferentes temos agora para implementar a operação de inversão em um circuito lógico?
2. Implemente a expressão $x = (A + B)(C + D)$ usando portas AND e OR. Agora implemente esta expressão utilizando apenas portas NOR. Para isso, converta cada porta AND e OR que seja necessária pela sua implementação em portas NOR, como visto na Fig. 3-30. Qual circuito é mais eficiente?
3. Escreva a expressão para a saída do circuito da Fig. 3-32(c) e use o teorema de DeMorgan para mostrar que esta é equivalente à expressão dada para o circuito da Fig. 3-32(a).

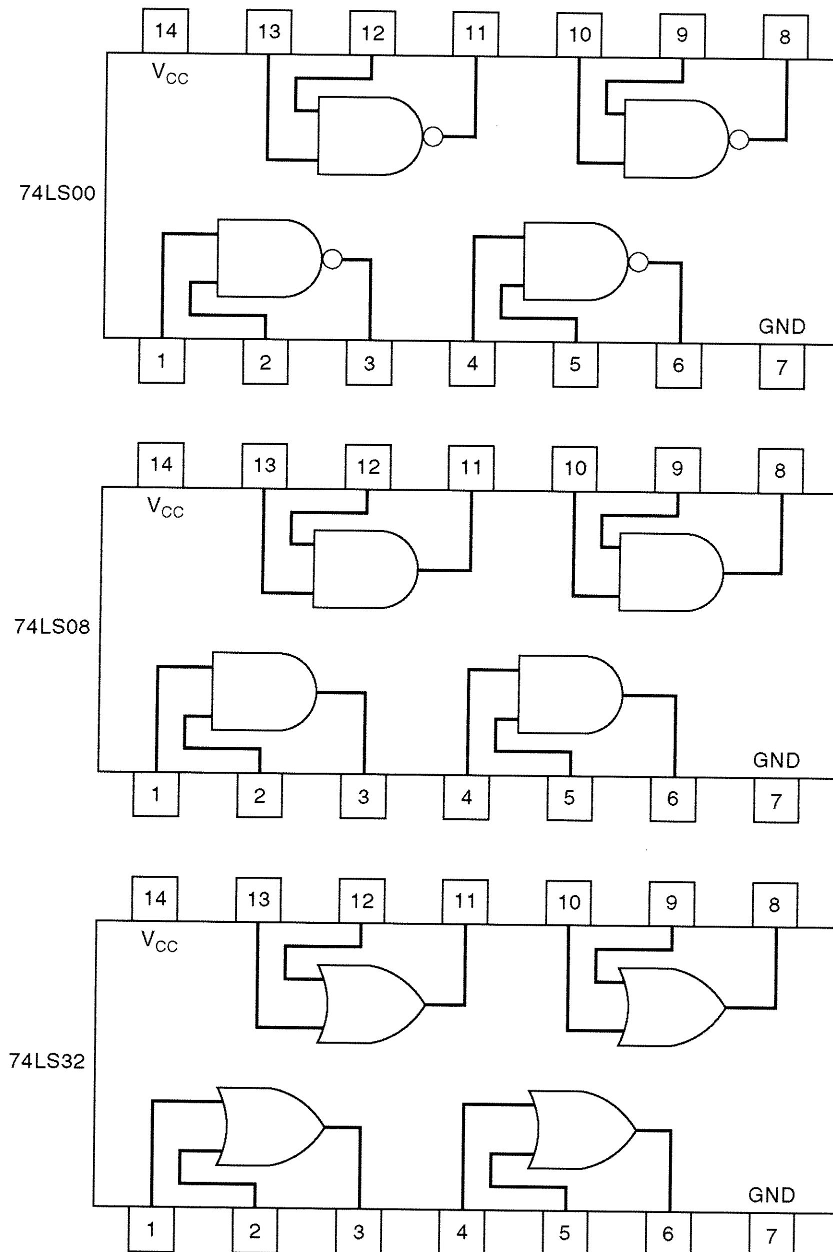


Fig. 3-31 CIs disponíveis para o Exemplo 3-18.

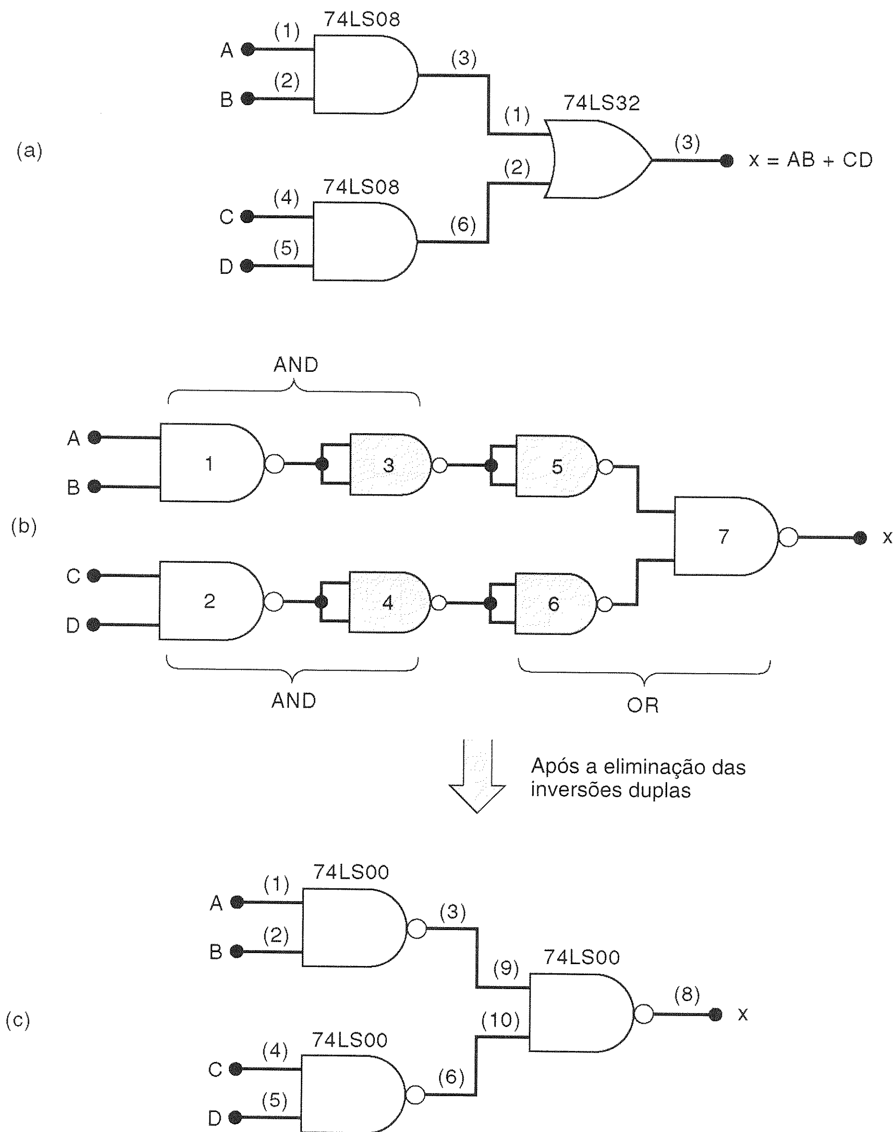


Fig. 3-32 Implementações possíveis para o Exemplo 3-18.

3-13 REPRESENTAÇÕES ALTERNATIVAS DAS PORTAS LÓGICAS

Introduzimos as cinco portas lógicas básicas (AND, OR, NOT, NAND e NOR) e os símbolos padronizados usados para representá-las em diagramas de circuitos lógicos. Embora você possa encontrar alguns diagramas de circuitos que ainda utilizam exclusivamente estes símbolos, é cada vez mais comum encontrarmos diagramas de circuitos que utilizam **símbolos lógicos alternativos em conjunto** com os símbolos padronizados.

Antes de discutirmos as razões para utilizar um símbolo alternativo para uma porta lógica, apresentaremos os símbolos alternativos para cada porta lógica e mostraremos que eles são equivalentes aos símbolos padronizados. Observe a Fig. 3-33. O lado esquerdo da figura mostra o símbolo padronizado para cada porta lógica, e o lado direito mostra os símbolos alternativos. O símbolo alternativo para cada porta é obtido a partir do símbolo original fazendo-se o seguinte:

1. Inverta cada entrada e saída do símbolo padronizado. Isto é feito adicionando bolhas (pequenos círculos) em entradas e saídas que não possuem bolhas e removendo-as de onde elas já existem.
2. Troque o símbolo da operação AND pelo símbolo da operação OR ou troque de OR para AND (no caso especial do INVERSOR, o símbolo da operação não é trocado).

Por exemplo, o símbolo padronizado NAND é o símbolo AND com uma bolha em sua saída. Seguindo os passos descritos anteriormente, temos que remover a bolha da saída e adicionar uma bolha para cada entrada. Feito isto, podemos trocar o símbolo AND pelo símbolo OR. O resultado será um símbolo OR com bolhas em suas entradas.

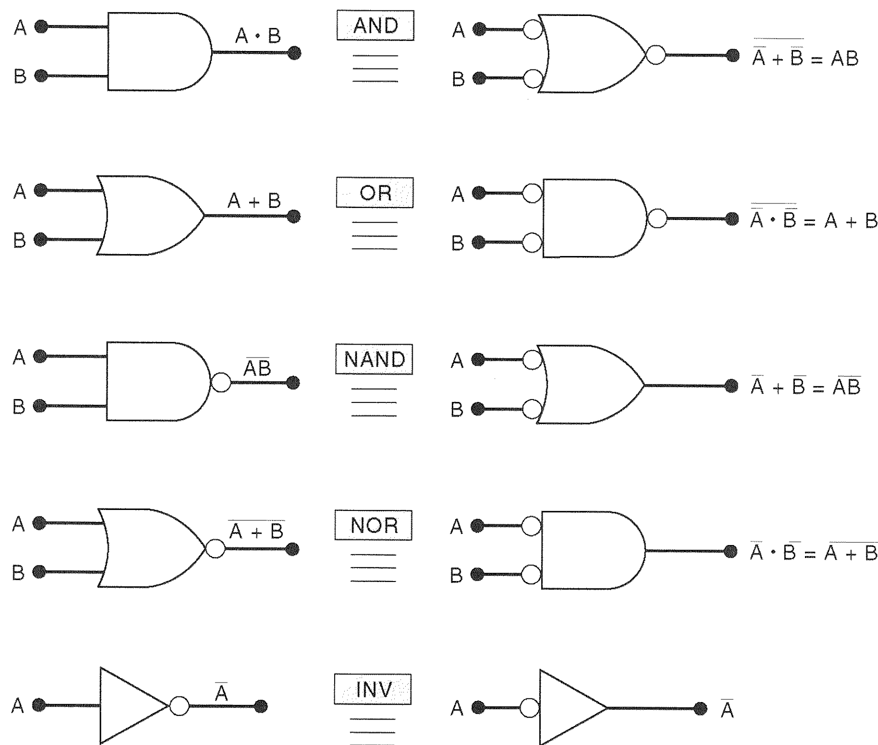


Fig. 3-33 Símbolos padronizados e alternativos para várias portas lógicas e para o inversor.

Podemos facilmente provar que esse símbolo alternativo é equivalente ao símbolo padronizado utilizando o teorema de DeMorgan e lembrando que a bolha representa uma operação de inversão. A expressão para a saída de um símbolo padronizado NAND é $\overline{AB} = \overline{A} + \overline{B}$, que é a mesma expressão para a saída do símbolo alternativo. Este mesmo procedimento pode ser seguido para cada um dos pares de símbolos da Fig. 3-33.

Vários pontos devem ser enfatizados no que se refere às equivalências de símbolos lógicos:

1. As equivalências podem ser estendidas a portas com *qualquer* número de entradas.
2. Nenhum dos símbolos padronizados tem bolhas em suas entradas, ao passo que todos os alternativos têm.
3. Os símbolos padronizados e alternativos para cada porta representam o mesmo circuito físico. *Não há diferenças nos circuitos representados pelos dois símbolos.*
4. NAND e NOR são portas inversoras, e portanto os símbolos padronizado e alternativo para cada uma terão uma bolha *ou* na entrada *ou* na saída. AND e OR são portas *não-inversoras*, e portanto os símbolos alternativos terão bolhas *tanto nas* entradas *quanto* na saída.

Interpretação dos Símbolos Lógicos

Cada um dos símbolos lógicos da Fig. 3-33 fornece uma interpretação única do funcionamento da porta. Antes de demonstrar estas interpretações, devemos primeiro estabelecer o conceito de **níveis lógicos ativos.**

Quando uma linha de entrada ou saída de um símbolo de circuito lógico *não possui a bolha de inversão*, diz-se que esta linha é ativa em nível lógico ALTO (**ativa-ALTO**). Quando a linha de entrada ou saída *possui a bolha de inversão*, diz-se que esta linha é ativa em nível lógico BAIXO (**ativa-BAIXO**). A presença ou ausência da bolha de inversão, então, determina a condição ativa-ALTO/ativa-BAIXO das entradas ou saídas e é usada para interpretar a operação do circuito.

Para ilustrar, a Fig. 3-34(a) mostra o símbolo padronizado para uma porta NAND. O símbolo tem uma bolha de inversão em sua saída e não possui bolhas nas entradas. Então ela possui uma saída ativa-BAIXO e entradas do tipo ativa-ALTO. A operação lógica representada por esse símbolo pode ser interpretada da seguinte maneira:

A saída vai para BAIXO somente quando *todas* as entradas estão em ALTO.

Observe que esta frase diz que a entrada irá para o seu estado ativo somente quando *todas* as entradas estiverem em seus estados ativos. A palavra "todas" é usada por causa do símbolo AND.

O símbolo alternativo para a porta NAND mostrado na Fig. 3-34(b) possui uma saída ativa-ALTO e entradas do tipo ativa-BAIXO. Assim, sua operação pode ser descrita como se segue:

A saída vai para ALTO quando *qualquer das entradas* estiver em BAIXO.

Esta afirmação nos diz que a saída estará no seu estado ativo sempre que *qualquer* uma das entradas estiver no seu estado ativo. A palavra "qualquer" é usada por causa do símbolo OR.

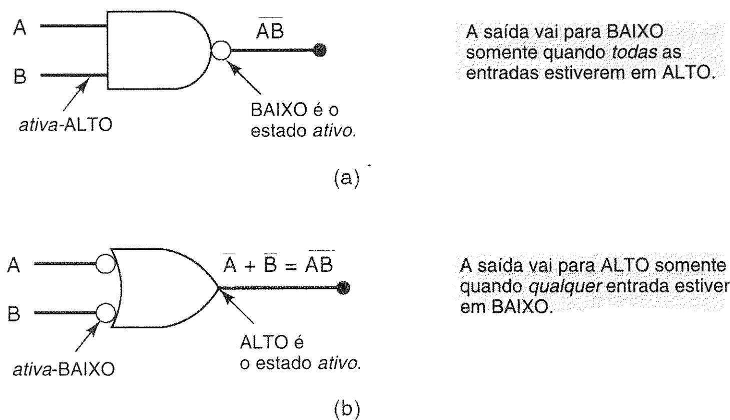


Fig. 3-34 Interpretação dos dois símbolos das portas NAND.

Com um pouco de raciocínio, podemos notar que essas duas interpretações para os símbolos da porta NAND na Fig. 3-34 são maneiras diferentes de dizer a mesma coisa.

Resumo

Neste ponto, você deve estar imaginando por que existe a necessidade de termos dois símbolos e interpretações diferentes para cada porta lógica. Felizmente, as razões disso ficarão claras após estudarmos a próxima seção. Por enquanto, vamos resumir pontos importantes referentes à representação de portas lógicas.

1. Para obter o símbolo alternativo para cada porta lógica, tome o símbolo padronizado e troque seu símbolo de operação (OR por AND, ou AND por OR) e troque as bolhas de inversão nas entradas e na saída (isto é, retire-as se estiverem presentes e coloque-as se não estiverem).
2. Para interpretar a operação da porta lógica, primeiro observe qual o estado lógico, 0 ou 1, é o estado ativo para as entradas e qual é o estado ativo para a saída. Feito isso, descubra se o estado ativo da saída é produzido quando *todas* as entradas estiverem no seu estado ativo (se o símbolo AND for usado) ou quando *qualquer* uma das entradas estiver no seu estado ativo (se o símbolo OR for usado).

EXEMPLO 3-19

Interprete os dois símbolos para a porta lógica OR.

Solução

As respostas estão mostradas na Fig. 3-35. Observe que a palavra "qualquer" é usada quando o símbolo de operação é o símbolo OR e a palavra "todas" é usada quando utilizamos o símbolo AND.

Questões de Revisão

1. Escreva a interpretação para a operação realizada pelo símbolo padronizado NOR na Fig. 3-33.
2. Repita a questão 1 para o símbolo alternativo da porta NOR.
3. Repita a questão 1 para o símbolo alternativo da porta AND.
4. Repita a questão 1 para o símbolo padronizado da porta AND.

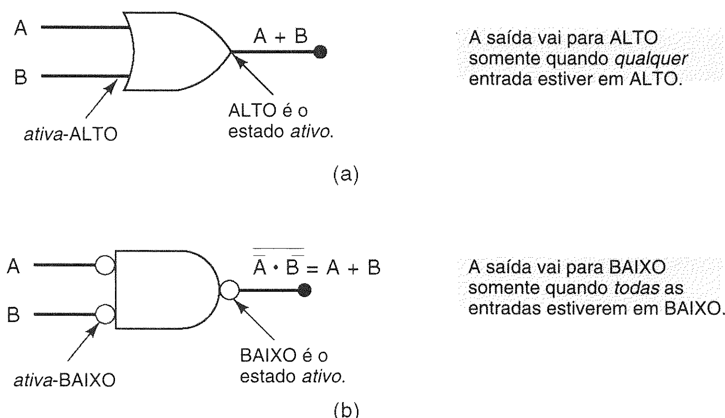


Fig. 3-35 Interpretação dos dois símbolos das portas OR.

3-14 QUE REPRESENTAÇÃO DE PORTA LÓGICA USAR

Alguns projetistas de circuitos lógicos e alguns livros utilizam apenas os símbolos padronizados para as portas lógicas nos esquemáticos de seus circuitos. Apesar de esta prática não estar incorreta, ela não torna a operação do circuito mais fácil de acompanhar. A utilização adequada dos símbolos alternativos para as portas nos diagramas de circuitos pode tornar a operação do circuito bem mais clara. Isto pode ser ilustrado considerando-se o exemplo a seguir da Fig. 3-36.

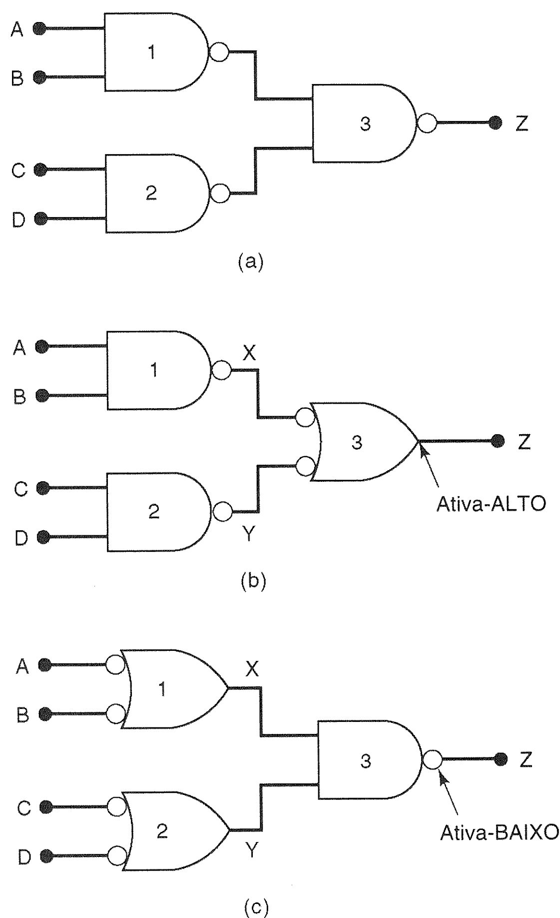
O circuito da Fig. 3-36(a) contém três portas NAND conectadas para produzir uma saída *Z* que depende das entradas *A*, *B*, *C* e *D*. O diagrama do circuito utiliza o símbolo padrão para cada porta NAND. Mesmo esse diagrama estando logicamente correto, ele não facilita no entendimento de como o circuito funciona. As representações do circuito apresentadas nas Fig. 3-36(b) e (c), no entanto, podem ser analisadas mais facilmente para determinar a operação do circuito.

A representação da Fig. 3-36(b) é obtida do diagrama do circuito original substituindo-se a porta NAND 3 pelo seu símbolo alternativo. Nesse diagrama, a saída *Z* é gerada de um símbolo de porta NAND que tem uma saída ativa em ALTO. Desse modo, podemos dizer que *Z* vai para ALTO quando ou *X* ou *Y* for BAIXO. Agora, já que *X* e *Y* aparecem na saída de símbolos NAND, que possuem saídas ativas em BAIXO, podemos dizer que *X* vai para BAIXO somente se $A = B = 1$ e *Y* vai para BAIXO somente se $C = D = 1$. Em resumo, podemos descrever a operação do circuito do seguinte modo:

A saída *Z* vai para ALTO sempre que ou $A = B = 1$ ou $C = D = 1$ (ou ambos).

Esta descrição pode ser colocada na forma de tabela-verdade fazendo $Z = 1$ para os casos em que $A = B = 1$ e para os casos em que $C = D = 1$. Para todos os outros casos, *Z* deve ser 0. A tabela-verdade resultante é mostrada na Fig. 3-36(d).

A representação da Fig. 3-36(c) é obtida do diagrama do circuito original substituindo-se as portas NAND 1 e 2 pelos seus símbolos alternativos. Nesta representação equivalente, a saída *Z* é gerada de uma porta NAND que tem uma saída ativa-BAIXO. Portanto, podemos dizer que *Z* vai para BAIXO somente quando $X = Y = 1$. Como *X* e *Y* são saídas ativas em ALTO, podemos dizer que *X* será ALTO quando ou *A* ou *B* for BAIXO e *Y* será ALTO quando ou *C* ou *D* for



A	B	C	D	Z
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

(d)

Fig. 3-36 (a) Circuito original utilizando símbolos padrões NAND; (b) representação equivalente onde a saída *Z* está ativa-ALTO; (c) representação equivalente onde a saída *Z* está ativa-BAIXO; (d) tabela-verdade.

BAIXO. Resumindo, podemos descrever a operação do circuito do seguinte modo:

A saída *Z* vai para BAIXO somente quando *A* ou *B* for BAIXO e *C* ou *D* for BAIXO.

Esta descrição pode ser colocada na forma de tabela-verdade tornando $Z = 0$ para todos os casos em que pelo menos uma das entradas *A* ou *B* está BAIXA, ao mesmo tempo em que pelo menos uma das entradas *C* ou *D* está BAIXA. Para todos os outros casos, *Z* deve ser 1. A tabela-verdade resultante é a mesma que foi obtida do diagrama do circuito da Fig. 3-36(b).

Que Diagrama de Circuito Deve Ser Usado?

A resposta para esta pergunta depende da função específica sendo realizada pela saída do circuito. Se o circuito está sendo usado para causar alguma ação (por exemplo: ligar um LED — *Light Emitting Diode* — ou ativar um outro circuito lógico) quando a saída *Z* vai para o estado 1, então dizemos que a saída *Z* deve ser ativa-ALTO, e o diagrama de circuito da Fig. 3-36(b) deveria ser usado. Por outro lado, se o circuito está sendo usado para causar alguma ação quando *Z* vai para o estado 0, então Z deve ser ativa-BAIXO, e o diagrama de circuito da Fig. 3-36(c) deveria ser usado.

Naturalmente existem situações em que *ambos* os estados de saída são usados para produzir diferentes ações, e qualquer um pode ser considerado o estado ativo. Para esses casos, qualquer representação de circuito pode ser usada.

Colocação da Bolha

Veja a representação do circuito da Fig. 3-36(b) e note que os símbolos para as portas NAND 1 e 2 foram escolhidos para terem saídas ativas em BAIXO, para combinar com as entradas ativas em BAIXO da porta NAND 3. Veja a representação do circuito da Fig. 3-36(c) e note que os símbolos para as portas NAND 1 e 2 foram escolhidos para terem saídas ativas em ALTO, para combinar com as entradas ativas em ALTO da porta NAND 3. Isto leva para a seguinte regra geral na preparação de esquemáticos de circuitos lógicos:

Sempre que possível, escolha símbolos para as portas tais que saídas com bolha sejam conectadas em entradas com bolha, e saídas sem bolha em entradas sem bolha.

Os exemplos a seguir mostram como essa regra pode ser aplicada.

EXEMPLO 3-20

O circuito lógico na Fig. 3-37(a) está sendo usado para ativar um alarme quando sua saída *Z* vai para ALTO. Modifique o diagrama do circuito de modo que ele represente mais eficientemente a operação do circuito.

Solução

Já que $Z = 1$ ativará o alarme, *Z* deve ser ativa-ALTO. Logo, o símbolo da porta AND 2 não deve ser mudado. O símbolo da porta NOR deveria ser substituído pelo símbolo alternativo sem bolha na saída (ativa em ALTO), para combinar com a entrada sem bolha da porta AND 2, conforme mostrado na Fig. 3-37(b). Note que o circuito agora tem saídas sem bolha conectadas nas entradas sem bolha da porta 2.

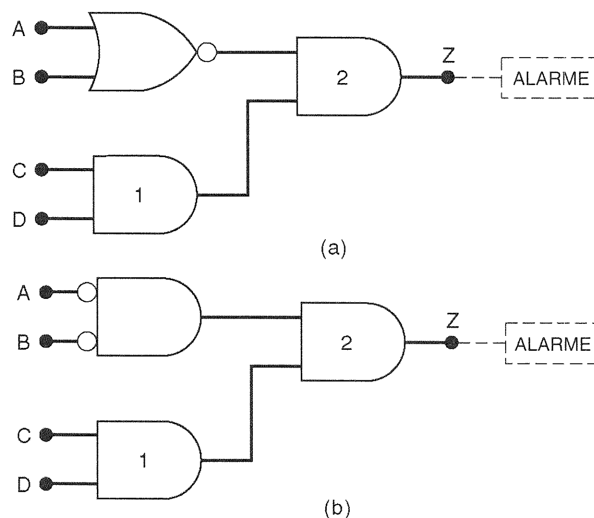


Fig. 3-37 Exemplo 3-20.

EXEMPLO 3-21

Quando a saída do circuito lógico na Fig. 3-38(a) vai para BAIXO, ela aciona um outro circuito lógico. Modifique o diagrama do circuito para representar mais eficientemente a operação do circuito.

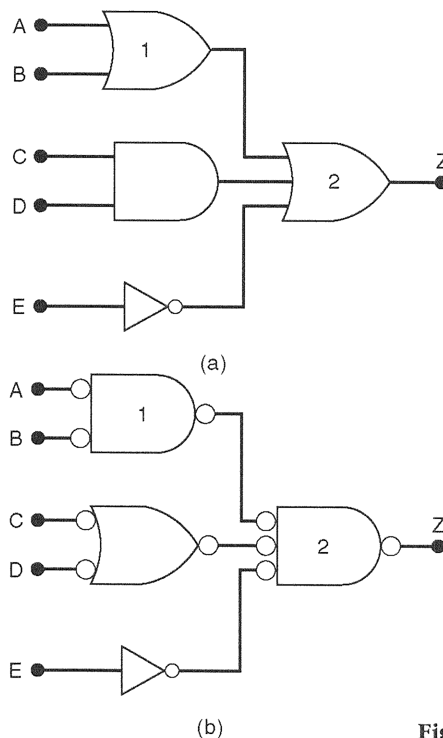


Fig. 3-38 Exemplo 3-21.

Solução

Já que Z deve ser ativa-BAIXO, o símbolo para a porta OR 2 deve ser mudado para o símbolo alternativo, como mostra a Fig. 3-38(b). O novo símbolo da porta OR 2 tem entradas com bolha, e portanto os símbolos da porta AND e da porta OR 1 devem ser trocados para terem saídas com bolhas, conforme mostra a Fig. 3-38(b). O INVERSOR já possui saída com bolha. Agora o circuito tem todas as saídas com bolha conectadas nas entradas com bolha da porta 2.

Analisando Circuitos

Quando um esquemático de circuito lógico é desenhado usando as regras que utilizamos nesses exemplos, é bem mais fácil para um engenheiro ou técnico (ou estudante) acompanhar o fluxo do sinal através do circuito e determinar as condições de entrada que são necessárias para ativar a saída. Isto será ilustrado nos próximos exemplos, que "por acaso" usam diagramas de circuitos obtidos de esquemáticos de um microcomputador real.

EXEMPLO 3-22

O circuito lógico na Fig. 3-39 gera uma saída $\overline{MEM}$, que é usada para ativar os CIs de memória de um microcomputador. Determine as condições de entrada necessárias para ativar $\overline{MEM}$.

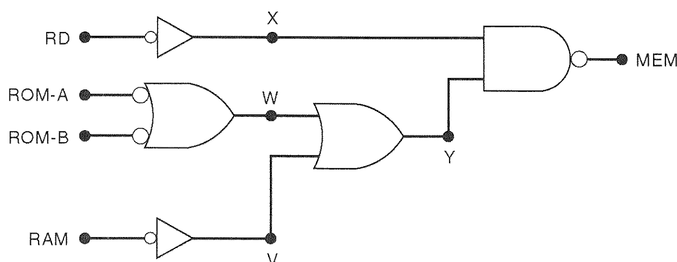


Fig. 3-39 Exemplo 3-22.

Solução

Uma maneira de fazer isso é escrever a expressão para $\overline{MEM}$ em termos das entradas $\overline{RD}$, $\overline{ROM\text{-}A}$, $\overline{ROM\text{-}B}$ e $\overline{RAM}$ e avaliá-la para as 16 combinações possíveis destas entradas. Apesar de esse método funcionar, ele demanda muito mais trabalho do que seria necessário.

Um método mais eficiente é interpretar o diagrama do circuito usando as idéias que desenvolvemos nas duas últimas seções. Os passos são os seguintes:

1. $\overline{MEM}$ é ativa-BAIXO, e fica BAIXO somente quando X e Y estão em ALTO.
2. X fica ALTO somente quando $\overline{RD} = 0$.
3. Y fica ALTO quando ou W ou V está em ALTO.
4. V fica ALTO quando $\overline{RAM} = 0$.
5. W fica ALTO quando ou $\overline{ROM\text{-}A}$ ou $\overline{ROM\text{-}B} = 0$.
6. Em resumo, $\overline{MEM}$ vai para BAIXO somente quando $\overline{RD} = 0$ e pelo menos uma das três entradas $\overline{ROM\text{-}A}$, $\overline{ROM\text{-}B}$ ou $\overline{RAM}$ estiver em BAIXO.

EXEMPLO 3-23

O circuito lógico na Fig. 3-40 é usado para controlar o motor de uma unidade de disco quando o microcomputador está enviando ou recebendo dados do disco. O circuito ligará o motor quando $DRIVE = 1$. Determine as condições de entrada necessárias para ligar o motor.

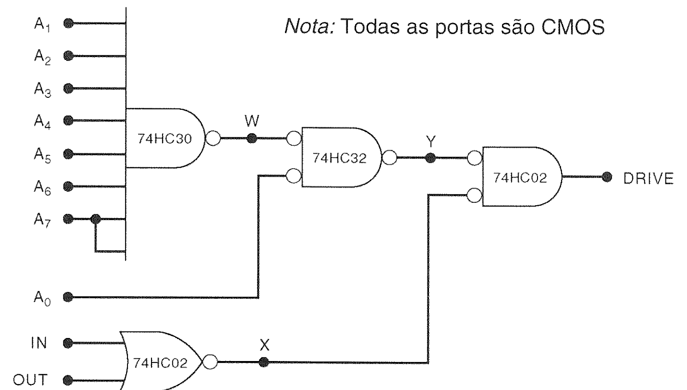


Fig. 3-40 Exemplo 3-23.

Solução

Mais uma vez, interpretaremos o diagrama passo a passo:

1. $DRIVE$ é ativa-ALTO, e vai para ALTO somente quando $X = Y = 0$.
2. X fica BAIXO quando ou IN ou OUT está em ALTO.
3. Y fica BAIXO somente quando $W = 0$ e $A_0 = 0$.
4. W fica BAIXO somente quando A_1 até A_7 estiverem em ALTO.
5. Em resumo, $DRIVE$ fica ALTO quando $A_1 = A_2 = A_3 = A_4 = A_5 = A_6 = A_7 = 1$ e $A_0 = 0$, e ou IN ou OUT ou ambos forem 1.

Note o símbolo diferente para a porta NAND CMOS de 8 entradas (74HC30); repare também que o sinal A_7 está conectado em duas entradas da NAND.

Níveis de Acionamento

Descrevemos sinais lógicos como estando ativos em BAIXO ou ativos em ALTO. Por exemplo, a saída $\overline{MEM}$ na Fig. 3-39 é ativa-BAIXO, e a saída $DRIVE$ na Fig. 3-40 é ativa-ALTO, tendo em vista que esses estados de saída fazem algo acontecer. Do mesmo modo, a Fig. 3-40 tem as entradas de A_1 até A_7 ativas em ALTO, e a entrada A_0 ativa em BAIXO.

Quando um sinal lógico está em seu estado ativo, pode-se dizer que ele está **acionado**. Por exemplo, quando dizemos que a entrada A_0 está acionada, estamos di-

zendo que ela está no seu estado ativo-BAIXO. Quando um sinal lógico não está no seu estado ativo, diz-se estar **não-acionado**. Portanto, quando dizemos que *DRIVE* está não-acionado, significa que ele está no seu estado inativo (BAIXO).

É claro que os termos "acionado" e "não-acionado" são sinônimos de "ativo" e "inativo", respectivamente:

acionado = ativo
não-acionado = inativo

Ambos os termos são de uso comum na área digital, portanto você deve reconhecer os dois modos de descrever o estado ativo de um sinal lógico.

Identificando Sinais Lógicos Ativos em BAIXO

Tornou-se prática comum usar uma barra sobreposta para identificar sinais ativos em BAIXO. A barra serve como outra indicação de que o sinal é ativo em BAIXO, e é claro que a ausência da barra significa que o sinal é ativo em ALTO.

Para ilustrar, todos os sinais na Fig. 3-39 são ativos em BAIXO e portanto podem ser identificados como segue:

$\overline{RD}$, $\overline{ROM\text{-}A}$, $\overline{ROM\text{-}B}$, $\overline{RAM}$, $\overline{MEM}$

Lembre-se, a barra é simplesmente um modo de enfatizar que esses sinais são ativos em BAIXO. Empregaremos essa convenção para identificação de sinais lógicos sempre que for apropriado.

Identificando Sinais de Dois Estados

Freqüentemente, um sinal de saída tem dois estados ativos, isto é, ele tem uma função importante no estado ALTO e uma outra no estado BAIXO. É usual identificar tais sinais de modo que ambos os estados ativos sejam aparentes. Um exemplo comum é o sinal de leitura/escrita $RD/\overline{WR}$, [do inglês, *read/write*], que é interpretado como segue: quando este sinal está em ALTO, a operação de leitura (*RD*) é realizada; quando está em BAIXO, a operação de escrita ($\overline{WR}$) é realizada.

Questões de Revisão

1. Use o método dos Exemplos 3-22 e 3-23 para determinar as condições de entrada necessárias para ativar a saída do circuito na Fig. 3-37(b).
2. Repita a questão 1 para o circuito da Fig. 3-38(b).
3. Quantas portas NAND existem na Fig. 3-39?
4. Quantas portas NOR existem na Fig. 3-40?
5. Qual será o nível de saída na Fig. 3-38(b) quando todas as entradas estiverem acionadas?
6. Que entradas são necessárias para acionar a saída de alarme na Fig. 3-37(b)?
7. Quais dos seguintes sinais são ativos em BAIXO: *RD*, $\overline{W}$ e $R/\overline{W}$?

3-15 SÍMBOLOS LÓGICOS DO PADRÃO IEEE/ANSI

Os símbolos lógicos que usamos em todo este capítulo são símbolos padronizados e bem conhecidos amplamente utilizados na indústria digital há muitos anos. Estes símbolos representam bem as portas lógicas básicas porque cada símbolo de porta tem uma forma característica, e cada entrada tem a mesma função. Eles não fornecem informação útil suficiente, no entanto, para dispositivos lógicos mais complexos tais como: flip-flops, contadores, decodificadores, multiplexadores, memórias e CIs de interface de microprocessadores. Estes CIs complexos freqüentemente têm várias entradas e saídas com diferentes funções e modos de operação.

Para fornecer informação mais útil sobre esses dispositivos lógicos complexos, um novo conjunto de símbolos padronizados foi introduzido em 1984 pelo Padrão IEEE/ANSI 91-1984. Esses novos **símbolos IEEE/ANSI** estão sendo, paulatinamente, aceitos por mais e mais companhias de eletrônica e fabricantes de CIs e já começam a aparecer nas suas literaturas técnicas. Além disso, os contratos militares nos Estados Unidos agora exigem a utilização desses novos símbolos. Por conseguinte, é importante ficar familiarizado com esses novos símbolos padronizados.

A principal diferença no padrão IEEE/ANSI é que ele utiliza símbolos retangulares para todos os dispositivos, em vez de formas diferentes de símbolos para cada dispositivo. Um sistema especial de *notação de dependência* é usado para indicar como as saídas *dependem* das entradas. A Fig. 3-41 mostra os novos símbolos retangulares lado a lado com

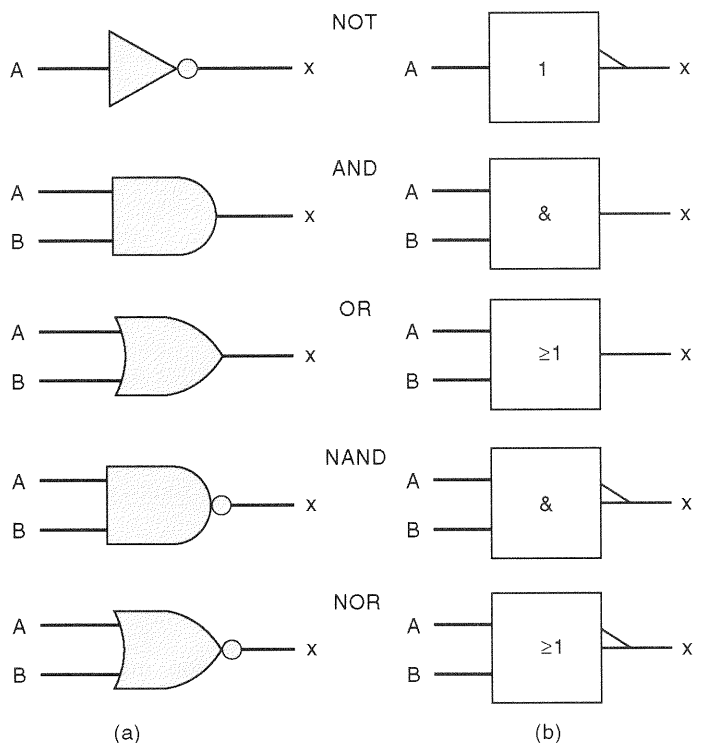


Fig. 3-41 Símbolos lógicos padronizados: (a) tradicionais; (b) retangulares.

os símbolos tradicionais para as portas básicas. Estude-os cuidadosamente e note os seguintes pontos:

1. Os símbolos retangulares usam um pequeno triângulo no lugar da pequena bolha dos símbolos tradicionais para indicar inversão do nível lógico. A presença ou ausência do triângulo também indica se uma entrada ou saída é ativa-BAIXO ou ativa-ALTO.
2. Uma notação especial dentro de cada símbolo retangular descreve a relação lógica entre entradas e saída. O "1" dentro do símbolo INVERSOR indica um dispositivo com somente *uma* entrada; o triângulo na saída significa que a saída vai para o seu estado ativo-BAIXO quando aquela única entrada estiver no seu estado ativo-ALTO. O "&" dentro do símbolo AND significa que a saída vai para o seu estado ativo-ALTO quando todas as entradas estiverem em seus estados ativos em ALTO. O "≥" dentro da porta OR significa que a saída vai para o seu estado ativo (ALTO) sempre que *uma ou mais* entradas estiverem no seu estado ativo (ALTO).
3. Os símbolos retangulares para NAND e NOR são os mesmos de AND e OR, respectivamente, apenas adicionando-se o pequeno triângulo de inversão na saída.

Símbolos IEEE/ANSI para CIs de Portas Lógicas

Os símbolos retangulares também podem ser usados para representar a lógica completa de um CI que contenha um certo número de portas independentes. Isto está ilustrado na Fig. 3-42 para o CI TTL INVERSOR sêxtuplo 74LS04, e na Fig. 3-43 para o CI CMOS NAND duplo de quatro entradas 74HC20. Cada porta lógica é representada como um bloco retangular separado. Repare que os símbolos retangulares indicam a notação da operação lógica apenas no bloco superior; entende-se que deve ser aplicada para todos os blocos representativos das outras portas no chip.

É importante compreender a diferença entre os dois modos alternativos para representar uma porta lógica num

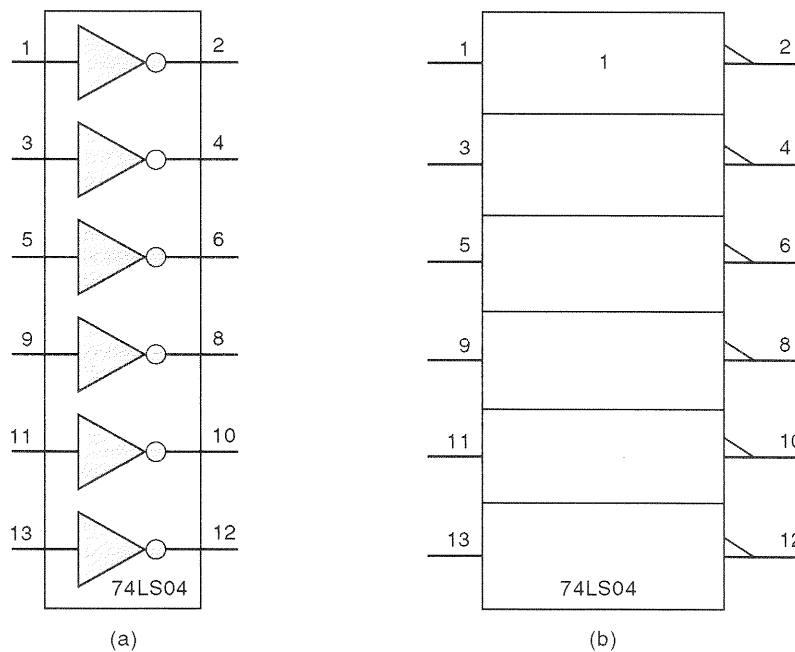


Fig. 3-42 CI INVERSOR sêxtuplo 74LS04: (a) símbolo lógico tradicional; (b) símbolo lógico retangular. A notação "1" aparece apenas no retângulo superior mas aplica-se a todos os outros blocos.

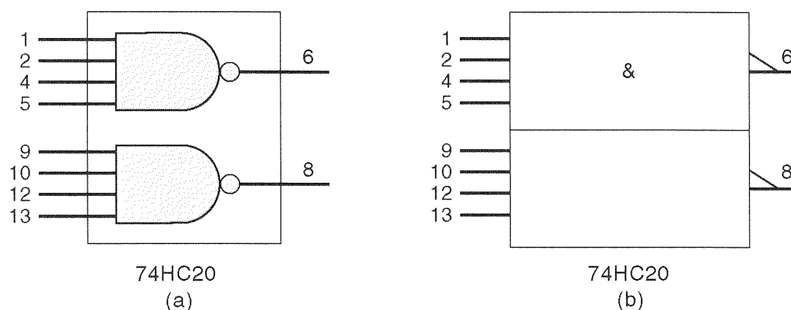


Fig. 3-43 CI NAND duplo de quatro entradas 74HC20: (a) símbolo tradicional; (b) símbolo retangular.

circuito e a diferença entre os dois padrões diferentes de símbolos para portas lógicas. Escolhe-se qual conjunto de símbolos padronizados usar — ou os símbolos tradicionais (formas características para cada tipo de porta) ou o novo padrão de símbolos retangulares. Independentemente do símbolo escolhido para utilização, existem duas maneiras possíveis de representar uma porta num diagrama de circuito dependendo do seu estado de saída ativo. Isto está ilustrado no Exemplo 3-24.

EXEMPLO 3-24

A Fig. 3-44(a) mostra as duas representações para uma porta NOR usando os símbolos lógicos tradicionais. Lembre-se, a escolha de que representação usar num diagrama de circuito é determinada pelo estado ativo desejado para a saída. Desenhe novamente as duas representações utilizando os novos símbolos IEEE/ANSI.

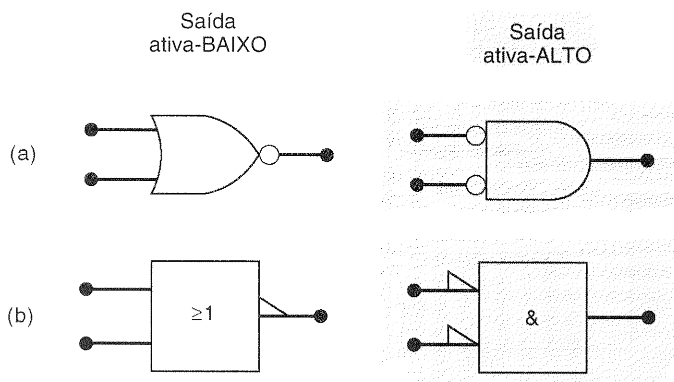


Fig. 3-44 Ambas as representações de uma porta NOR usando os dois tipos de símbolos: (a) tradicional; (b) retangular.

Solução

A Fig. 3-44(b) apresenta os resultados.

Símbolos IEEE/ANSI para CIs Complexos

Não haveria qualquer vantagem real para os novos símbolos se tivéssemos apenas que lidar com as portas lógicas básicas. Para os dispositivos lógicos mais complexos, entretanto, os novos símbolos padronizados com sua *notação de dependência* especificam a operação lógica completa do dispositivo. Isto torna praticamente desnecessário consultar o manual do fabricante para descobrir como um determinado CI lógico está funcionando em um circuito. Veremos exemplos disso quando encontrarmos os circuitos lógicos mais complexos em capítulos posteriores.

Os símbolos lógicos tradicionais são empregados na maioria dos diagramas de circuito em todo este livro, e os símbolos IEEE/ANSI são usados apenas ocasionalmente. Alguns problemas do final do capítulo necessitam de análise e construção de circuitos utilizando a notação mais nova. Além disso, sempre que um novo tipo de dispositivo lógico ou circuito for introduzido, ambos os tipos de símbolos serão apresentados. Desse modo, você ficará familiarizado com a notação de dependência que é a principal vantagem do novo padrão.

Questões de Revisão

1. Qual é a maior vantagem dos novos símbolos IEEE/ANSI?
2. Desenhe todas as portas lógicas básicas usando tanto os símbolos tradicionais quanto os símbolos padronizados IEEE/ANSI.
3. Repita a questão 2 para a representação alternativa de cada porta.

RESUMO

1. A álgebra booleana é uma ferramenta matemática usada em análise e projeto de circuitos digitais.
2. As operações booleanas básicas são as operações OR, AND e NOT.
3. Uma porta OR produz uma saída em ALTO quando qualquer entrada está em ALTO. Uma porta AND produz uma saída em ALTO somente quando todas as entradas estão em ALTO. Um circuito NOT (INVERSOR) produz uma saída que é o nível lógico oposto ao da entrada.
4. Uma porta NOR é o mesmo que uma porta OR com a saída conectada a um INVERSOR. Uma porta NAND é o mesmo que uma porta AND com a saída conectada a um INVERSOR.
5. Teoremas e regras booleanas podem ser usados para simplificar a expressão de um circuito lógico e podem levar a um modo mais simples de implementar o circuito.
6. Portas NAND podem ser usadas para implementar qualquer das operações booleanas básicas. Portas NOR podem ser usadas com o mesmo objetivo.
7. Tanto os símbolos padronizados quanto os alternativos podem ser usados para cada porta lógica, dependendo se a saída deve estar ativa-ALTO ou ativa-BAIXO.
8. O padrão IEEE/ANSI para símbolos lógicos utiliza símbolos retangulares para cada dispositivo lógico, com notações especiais dentro dos retângulos para mostrar como as saídas dependem das entradas.

TERMOS IMPORTANTES*

álgebra booleana
nível lógico
tabela-verdade
operação OR
porta OR
operação AND
porta AND
operação NOT
inversão-complemento
circuito NOT (INVERSOR)

*Esses termos podem ser encontrados em negrito no capítulo e estão definidos no Glossário ao final do livro.

porta NOR
porta NAND
teoremas booleanos
teoremas de DeMorgan
símbolos lógicos alternativos
níveis lógicos ativos
ativa(o)-ALTO
ativa(o)-BAIXO
acionado
não-acionado
símbolos IEEE/ANSI

PROBLEMAS

As letras em negrito que precedem alguns problemas são usadas para indicar a natureza ou tipo de problema como segue:

C (do inglês, *challenging*) problema desafiador
T (do inglês, *troubleshooting*) problema de depuração
D (do inglês, *design*) problema de projeto ou modificação de circuito
N (do inglês, *new concept*) novo conceito ou técnica não abordada no texto

SEÇÃO 3-3

3-1. Desenhe a forma de onda de saída para o circuito da Fig. 3-45.

3-2. Suponha que a entrada *A* na Fig. 3-45 seja involuntariamente colocada em curto com a terra (isto é, $A = 0$). Desenhe a forma de onda resultante na saída.

3-3. Suponha que a entrada *A* na Fig. 3-45 seja involuntariamente colocada em curto com a fonte de +5 V (isto é, $A = 1$). Desenhe a forma de onda resultante na saída.

3-4. Leia as afirmações a seguir relativas a uma porta OR. Inicialmente elas podem parecer válidas, mas após alguma análise você deve perceber que nenhuma é *sempre* verdadeira. Prove isto mostrando um exemplo específico para refutar cada afirmação.

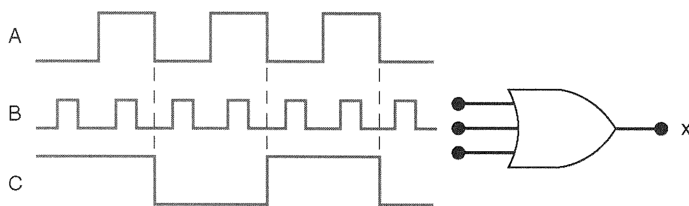


Fig. 3-45

(a) Se a forma de onda de saída de uma porta OR é a mesma forma de onda de uma das entradas, a outra entrada está sendo mantida permanentemente em BAIXO.
(b) Se a forma de onda de saída de uma porta OR está sempre em ALTO, uma das entradas está sendo permanentemente mantida em ALTO.

3-5. Quantos conjuntos diferentes de condições de entrada produzem uma saída em ALTO para uma porta OR de cinco entradas?

SEÇÃO 3-4

3-6. Troque a porta OR na Fig. 3-45 por uma porta AND.
(a) Desenhe a forma de onda de saída.
(b) Desenhe a forma de onda de saída se a entrada *A* está permanentemente colocada em curto com a terra.
(c) Desenhe a forma de onda de saída se a entrada *A* está permanentemente colocada em curto com +5 V.

D
3-7. Consulte a Fig. 3-4. Modifique o circuito de modo que o alarme seja ativado somente quando a pressão e a temperatura excederem os seus limites máximos ao mesmo tempo.

3-8. Troque a porta OR na Fig. 3-6 para uma porta AND e desenhe a forma de onda de saída.

3-9. Suponha que você tenha uma porta desconhecida de duas entradas que é ou uma porta OR ou uma porta AND. Que combinação de níveis de entrada você deve aplicar nas entradas da porta para determinar qual é o tipo da porta?

3-10. *Verdadeiro ou falso:* Não importa quantas entradas tenha, uma porta AND produz uma saída em ALTO para somente uma combinação dos níveis de entrada.

SEÇÕES 3-5 A 3-7

3-11. Acrescente um INVERSOR na saída da porta OR da Fig. 3-45. Desenhe a forma de onda na saída do INVERSOR.

3-12. (a) Escreva a expressão booleana para a saída *x* na Fig. 3-46(a). Determine o valor de *x* para todas as condições de entrada possíveis e relacione os valores em uma tabela-verdade.
(b) Repita para o circuito na Fig. 3-46(b).

3-13. Monte a tabela-verdade completa para o circuito da Fig. 3-15(b) determinando os níveis lógicos presentes em cada saída de porta para cada uma das 32 combinações possíveis de entrada.

3-14. Troque cada OR por um AND e cada AND por um OR na Fig. 3-15(b). Escreva a expressão para a saída.

3-15. Monte a tabela-verdade completa para o circuito da Fig. 3-16 determinando os níveis lógicos presentes em cada saída de porta para cada uma das 16 combinações possíveis de níveis de entrada.

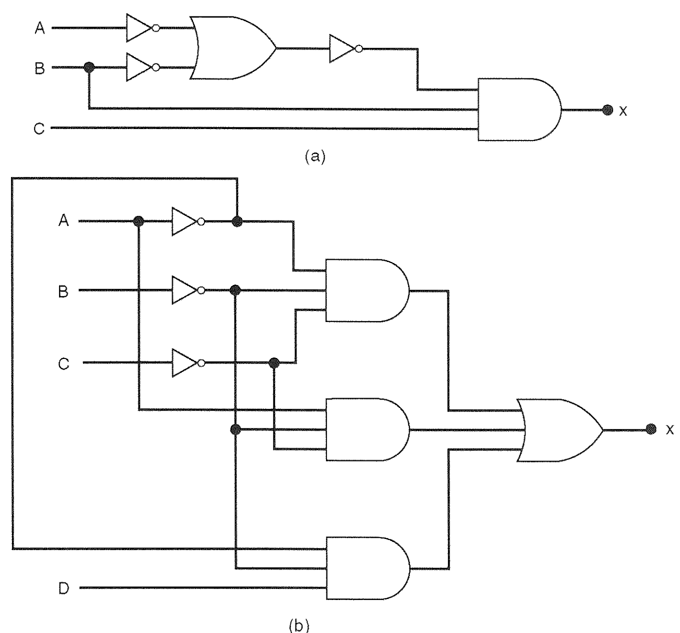


Fig. 3-46

SEÇÃO 3-8

3-16. Para cada uma das seguintes expressões, construa o circuito lógico correspondente, usando portas AND, OR e INVERSORes.

(a) $x = \overline{AB(C + D)}$

(b) $z = \overline{(A + B + \overline{C}D\overline{E})} + \overline{B}C\overline{D}$

(c) $y = (\overline{M + N} + \overline{P}Q)$

(d) $x = \overline{W + P\overline{Q}}$

(e) $z = MN(P + \overline{N})$

SEÇÃO 3-9

3-17. (a) Aplique as formas de onda de entrada da Fig. 3-47 numa porta NOR e desenhe a forma de onda de saída.

(b) Repita com *C* mantido permanentemente em BAIXO.

(c) Repita com *C* mantido ALTO.

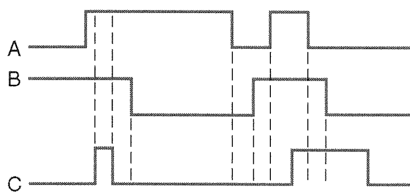


Fig. 3-47

3-18. Repita o Problema 3-17 para uma porta NAND.

3-19. Escreva a expressão de saída para o circuito da Fig. 3-48. Monte uma tabela-verdade completa.

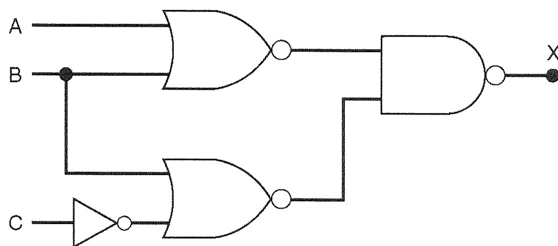


Fig. 3-48

3-20. Determine a tabela-verdade para o circuito da Fig. 3-24.

3-21. Modifique os circuitos que foram construídos no Problema 3-16 de modo que portas NAND e portas NOR sejam usadas sempre que for apropriado.

SEÇÃO 3-10

3-22. QUESTÃO DE FIXAÇÃO
Complete cada expressão.

(a) $A + 1 =$ ____________

(b) $A \cdot A =$ ____________

(c) $B \cdot \overline{B} =$ ____________

(d) $C + C =$ ____________

(e) $x \cdot 0 =$ ____________

(f) $D \cdot 1 =$ ____________

(g) $D + 0 =$ ____________

(h) $C + \overline{C} =$ ____________

(i) $G + GF =$ ____________

(j) $y + \overline{w}y =$ ____________

3-23. (a) Prove o teorema (15) experimentando todos os casos possíveis.

(b) Prove-o usando o teorema (14) para substituir *x*.

C

3-24.(a) Simplifique a expressão seguinte usando os teoremas (13b), (3) e (4):

$$x = (M + N)(\overline{M} + P)(\overline{N} + \overline{P})$$

(b) Simplifique a expressão seguinte usando os teoremas (13a), (8) e (6):

$$z = \overline{A}B\overline{C} + AB\overline{C} + B\overline{C}D$$

SEÇÕES 3-11 E 3-12

3-25. Prove os teoremas de DeMorgan experimentando todos os casos possíveis.

3-26. Simplifique cada uma das expressões seguintes utilizando os teoremas de DeMorgan.

(a) $\overline{\overline{A}B\overline{C}}$

(b) $\overline{\overline{A} + \overline{B}C}$

(c) $\overline{AB\overline{CD}}$

(d) $\overline{A\overline{(B + \overline{C})}D}$

(e) $\overline{(M + \overline{N})(\overline{M} + N)}$

(f) $\overline{\overline{\overline{AB}C}D}$

3-27. Use os teoremas de DeMorgan para simplificar a expressão para a saída da Fig. 3-48.

C

3-28. Converta o circuito da Fig. 3-46(b) para outro que use apenas portas NAND. Depois escreva a expressão de saída para o novo circuito, simplifique-a usando os teoremas de DeMorgan e compare-a com a expressão para o circuito original.

3-29. Converta o circuito da Fig. 3-46(a) para outro que use apenas portas NOR. Depois escreva a expressão para o novo circuito, simplifique-a usando os teoremas de DeMorgan e compare-a com a expressão para o circuito original.

3-30. Mostre como uma porta NAND de duas entradas pode ser construída com portas NOR de duas entradas.

3-31. Mostre como uma porta NOR de duas entradas pode ser construída com portas NAND de duas entradas.

3-32. Um avião a jato emprega um sistema para monitoração dos valores de rpm, pressão e temperatura dos motores utilizando sensores que operam como segue:

saída do sensor *RPM* = 0 somente quando a velocidade < 4800 rpm
saída do sensor *P* = 0 somente quando a pressão $< 1{,}5 \times 10^6$ N/m²
saída do sensor *T* = 0 somente quando a temperatura < 95ºC

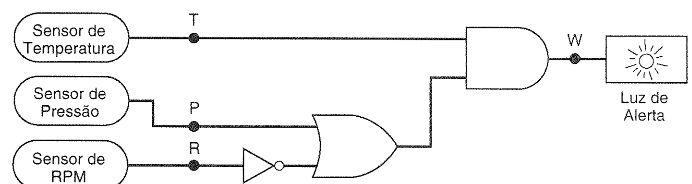


Fig. 3-49

A Fig. 3-49 mostra o circuito lógico que controla a luz de alerta da cabine do piloto para certas combinações das condições do motor. Suponha que um nível ALTO na saída *W* ativa a luz de alerta.

(a) Determine que condições do motor darão um alerta para o piloto.

(b) Altere o circuito para um outro que use apenas portas NAND.

SEÇÕES 3-13 E 3-14

3-33. Desenhe as representações padronizadas para cada porta lógica básica. Depois desenhe as representações alternativas.

3-34. Para cada sentença a seguir, desenhe a representação de porta lógica apropriada e indique o tipo de porta.

(a) Uma saída em ALTO ocorre apenas quando todas as três entradas estão em BAIXO.

(b) Uma saída em BAIXO ocorre quando qualquer uma das quatro entradas está em BAIXO.

(c) Uma saída em BAIXO ocorre apenas quando todas as oito entradas estão em ALTO.

3-35. O circuito da Fig. 3-48 é uma simples tranca de combinação digital cuja saída gera um sinal ativo-BAIXO $\overline{UNLOCK}$ para apenas uma combinação das entradas.

(a) Modifique o diagrama do circuito de modo que ele represente mais efetivamente a operação do circuito.

(b) Use o novo diagrama do circuito para determinar a combinação de entrada que ativa a saída. Faça isto analisando desde a saída usando as informações dadas pelos símbolos das portas como foi feito nos Exemplos 3-22 e 3-23. Compare os resultados com a tabela-verdade obtida no Problema 3-19.

3-36. (a) Determine as condições de entrada necessárias para ativar a saída *Z* na Fig. 3-37(b). Faça isto analisando desde a saída como foi feito nos Exemplos 3-22 e 3-23.

(b) Admita que é o estado BAIXO de *Z* que ativa o alarme. Altere o diagrama do circuito para refletir isto, e depois use o diagrama revisado para determinar as condições de entrada necessárias para ativar o alarme.

D

3-37. Modifique o circuito da Fig. 3-40 de modo que $A_1 = 0$ seja necessário para produzir *DRIVE* = 1 em vez de $A_1 = 1$.

3-38. Determine as condições de entrada necessárias para que a saída na Fig. 3-50 vá para o seu estado ativo.

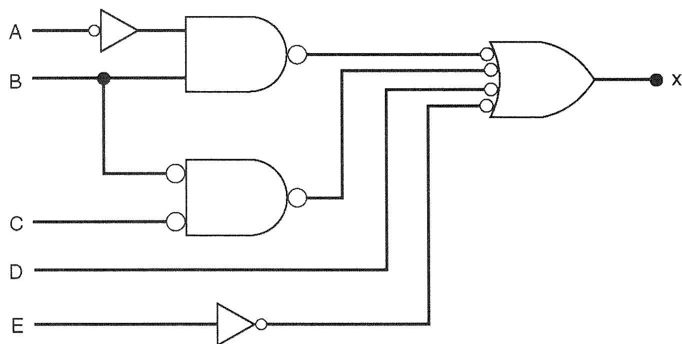


Fig. 3-50

3-39. Use os resultados do Problema 3-38 para obter a tabela-verdade completa para o circuito da Fig. 3-50.

3-40. Qual é o estado ativo para a saída da Fig. 3-50? E para a saída da Fig. 3-36(c)?

3-41. A Fig. 3-51 mostra uma aplicação de portas lógicas que simula os interruptores que usamos em nossas casas para acender e apagar uma luz de dois lugares diferentes. Aqui a luz é um LED que será LIGADO (conduzindo) quando a saída da porta NOR estiver em BAIXO. Note que esta saída é denominada $\overline{LIGHT}$ para indicar que é ativa-BAIXO. Determine as condições de entrada necessárias para ligar o LED. Depois verifique que o circuito opera como os interruptores descritos usando as chaves *A* e *B*. No Cap. 4 você aprenderá como projetar circuitos como este para produzir uma determinada relação entre entradas e saídas.

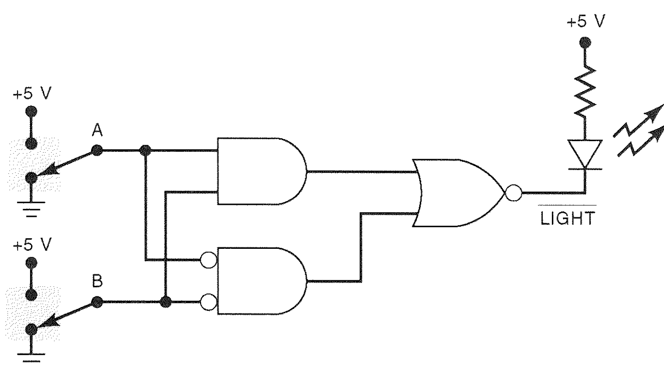


Fig. 3-51

SEÇÃO 3-15

3-42. Desenhe os circuitos da **(a)** Fig. 3-50 e **(b)** Fig. 3-51 usando os símbolos IEEE/ANSI.

3-43. Determine a expressão booleana para a saída *Z* na Fig. 3-52.

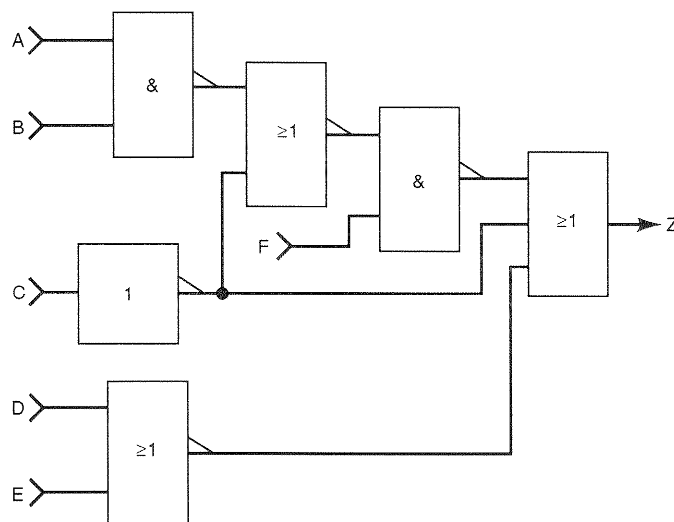


Fig. 3-52

C

3-44. Supõe-se que a saída do circuito da Fig. 3-52 é ativa-BAIXO. Desenhe-o para representar mais efetivamente a operação do circuito.

C

3-45. Use a versão redesenhada do circuito da Fig. 3-52 e faça o seguinte:

(a) Determine as várias condições de entrada que produzem um estado de saída ativa-BAIXO. Faça isto usando apenas o diagrama do circuito sem escrever a expressão para *Z* e sem gerar uma tabela-verdade completa. Os resultados deveriam ser:

A	B	C	D	E	F
1	1	1	1	1	1
1	1	1	1	0	1
1	1	1	0	1	1

(b) Verifique que a expressão simplificada para a saída Z é dada por

$$Z = \overline{ABCF(D + E)}$$

(c) Teste cada conjunto de condições de (a) na expressão obtida em (b) e verifique que cada uma produz $Z = 0$.

APLICAÇÃO EM MICROCOMPUTADOR

C

3-46. Consulte a Fig. 3-40 no Exemplo 3-23. As entradas A_7 até A_0 são entradas de *endereço* que são fornecidas para esse circuito por saídas do chip do microprocessador dentro do microcomputador. O código de endereço de oito bits de A_7 até A_0 seleciona qual dispositivo o microprocessador deseja ativar. No Exemplo 3-23, o código de endereço necessário para ativar a unidade de disco é A_7 até $A_0 = 11111110_2 = FE_{16}$.

Modifique o circuito de modo que o microprocessador deva fornecer um código de endereço de $4A_{16}$ para ativar a unidade de disco.

EXERCÍCIOS DESAFIADORES

C

3-47. Mostre como $x = AB\overline{C}$ pode ser implementado com uma porta NOR de duas entradas e uma porta NAND de duas entradas.

C

3-48. Implemente $y = ABCD$ usando portas NAND de duas entradas.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 3-2

1. $x = 1$ **2.** $x = 0$ **3.** 32

SEÇÃO 3-3

1. Todas as entradas em BAIXO **2.** $x = A + B + C + D + E + F$
3. Constante ALTO

SEÇÃO 3-4

1. Todas as cinco entradas = 1. **2.** Uma entrada em BAIXO manterá a saída em BAIXO.
3. Falso; veja a tabela-verdade de cada porta.

SEÇÃO 3-5

1. A saída do segundo INVERSOR será a mesma que a entrada A.
2. y será BAIXO somente para $A = B = 1$.

SEÇÃO 3-6

1. $x = \overline{A} + B + C + \overline{AD}$

SEÇÃO 3-7

1. $x = 1$ **2.** $x = 1$

SEÇÃO 3-8

1. Veja a Fig. 3-15(a). **2.** Veja a Fig. 3-17(b).
3. Veja a Fig. 3-15(b).

SEÇÃO 3-9

1. Todas as entradas em BAIXO **2.** $x = 0$
3. $x = \overline{A + B + \overline{C}\overline{D}}$

SEÇÃO 3-10

1. $y = A\overline{C}$
2. $y = \overline{A}\,\overline{B}\,\overline{D}$

SEÇÃO 3-11

1. $z = \overline{A}\,\overline{B} + C$ **2.** $y = (\overline{R} + S + \overline{T})Q$ **3.** O mesmo que a Fig. 3-28 exceto que o NAND é trocado por NOR. **4.** $y = A\overline{B}(C + \overline{D})$

SEÇÃO 3-12

1. Três **2.** O circuito NOR é mais eficiente porque pode ser implementado com um CI 74LS02. **3.** $x = (\overline{AB})(\overline{C}\,\overline{D}) = (\overline{\overline{AB}}) + (\overline{\overline{CD}}) = AB + CD$

SEÇÃO 3-13

1. Saída fica BAIXO quando qualquer entrada está em ALTO.
2. Saída fica ALTO somente quando todas as entradas estão em BAIXO.
3. Saída fica BAIXO quando qualquer entrada está em BAIXO.
4. Saída fica ALTO somente quando todas as entradas estão em ALTO.

SEÇÃO 3-14

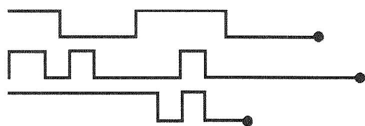
1. Z fica ALTO quando $A = B = 0$ e $C = D = 1$. **2.** Z fica BAIXO quando $A = B = 0$, $E = 1$, e ou C ou D ou ambos são 0.
3. Duas **4.** Duas **5.** BAIXO **6.** $A = B = 0$, $C = D = 1$ **7.** $\overline{W}$

SEÇÃO 3-15

1. Os símbolos IEEE/ANSI com sua notação de dependência especificam a operação completa do dispositivo lógico. **2.** Veja a Fig. 3-41. **3.** Veja a Fig. 3-44.

CAPÍTULO 4

Circuitos Lógicos Combinacionais



SUMÁRIO

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- ■ Converter uma expressão lógica na forma padrão do tipo soma-de-produtos.
- ■ Realizar os passos necessários para obter uma expressão do tipo soma-de-produtos com o objetivo de projetar um circuito lógico correspondente na sua forma mais simples.
- ■ Utilizar a álgebra booleana e o mapa de Karnaugh como ferramentas para simplificação e projeto de circuitos lógicos.
- ■ Explicar o funcionamento dos circuitos *Exclusive*-OR e *Exclusive*-NOR.
- ■ Projetar circuitos lógicos simples sem o auxílio da tabela-verdade.
- ■ Implementar circuitos de habilitação.
- ■ Citar as características básicas de CIs digitais.
- ■ Compreender as diferenças de operação existentes entre circuitos TTL e CMOS.
- ■ Utilizar regras básicas para pesquisa de falhas em sistemas digitais.
- ■ Deduzir, a partir de resultados de medidas, as falhas de funcionamento em circuitos lógicos combinacionais.
- ■ Descrever o princípio fundamental da lógica programável.

■ INTRODUÇÃO

No Cap. 3, estudamos a operação de todas as portas lógicas básicas e utilizamos a álgebra booleana para descrever e analisar circuitos que foram feitos a partir da combinação de portas lógicas. Estes circuitos podem ser classificados como circuitos lógicos *combinacionais* porque, em qualquer instante de tempo, o nível lógico da saída do circuito depende da combinação dos níveis lógicos presentes nas entradas. Um circuito combinacional não possui *memória*, e portanto sua saída depende apenas dos valores atuais das entradas.

Neste capítulo continuaremos nosso estudo de circuitos combinacionais, começando por um aprofundamento na simplificação de circuitos lógicos. Dois métodos serão usados: o primeiro utilizará os teoremas da álgebra booleana, e o segundo utilizará uma técnica de *mapeamento*. Além disso, iremos estudar técnicas simples para projetar circuitos lógicos que satisfaçam um dado conjunto de requisitos. Um estudo completo sobre o projeto de circuitos lógicos não é um dos nossos objetivos, mas os métodos que estudaremos proporcionarão uma boa introdução a este assunto.

A última parte deste capítulo trata da pesquisa de falhas em circuitos combinacionais. Esta primeira exposição sobre pesquisa de falhas deve ajudá-lo a desenvolver a capacidade de análise necessária para ser bem-sucedido nesta atividade. De modo a tornar este material o mais prático possível, primeiro introduziremos algumas características básicas de circuitos integrados de portas lógicas das famílias TTL e CMOS, juntamente com uma descrição dos tipos de falhas mais freqüentemente encontrados em circuitos digitais.

4-1 FORMA DE SOMA-DE-PRODUTOS

Os métodos de simplificação e projeto de circuitos lógicos que estudaremos exigem que a expressão esteja na forma de **soma-de-produtos.** Alguns exemplos de expressões deste tipo podem ser vistos a seguir:

1. $ABC + \overline{A}B\overline{C}$
2. $AB + \overline{A}B\overline{C} + \overline{C}\,\overline{D} + D$
3. $\overline{A}B + C\overline{D} + EF + GK + H\overline{L}$

Cada uma destas expressões do tipo soma-de-produtos consiste em dois ou mais termos AND (produtos) que por sua vez são conectados a uma porta OR. Cada termo AND consiste em uma ou mais variáveis que aparecem *individualmente* na sua forma complementada ou não. Por exemplo, na expressão $ABC + \overline{A}B\overline{C}$, o primeiro produto AND contém as variáveis A, B e C na sua forma não-complementada (não-invertida). O segundo produto contém A e C na sua forma complementada (invertida). Observe que em uma expressão do tipo soma-de-produtos, um sinal de inversão não pode cobrir mais do que uma variável em um termo (por exemplo, não poderíamos ter $\overline{ABC}$ ou $\overline{RST}$).

Produto-de-Somas

Uma outra forma geral para expressões lógicas às vezes é utilizada no projeto de circuitos lógicos. Ela é chamada de forma de **produto-de-somas**, e consiste em dois ou mais termos OR (somas) que por sua vez são conectados às entradas de uma porta AND. Cada termo OR contém uma ou mais variáveis na sua forma complementada ou não. A seguir, podemos ver algumas expressões do tipo produto-de-somas:

1. $(A + \overline{B} + C)(A + C)$
2. $(A + \overline{B})(\overline{C} + D)F$
3. $(A + C)(B + \overline{D})(\overline{B} + C)(A + \overline{D} + \overline{E})$

Os métodos de simplificação e projeto que serão usados são baseados em expressões do tipo soma-de-produtos, e portanto não utilizaremos muito a forma produto-de-somas. Ela, entretanto, aparecerá em alguns circuitos que têm uma estrutura particular.

Questões de Revisão

1. Quais das expressões a seguir estão na forma de soma-de-produtos?

   (a) $AB + CD + E$

   (b) $AB(C + D)$

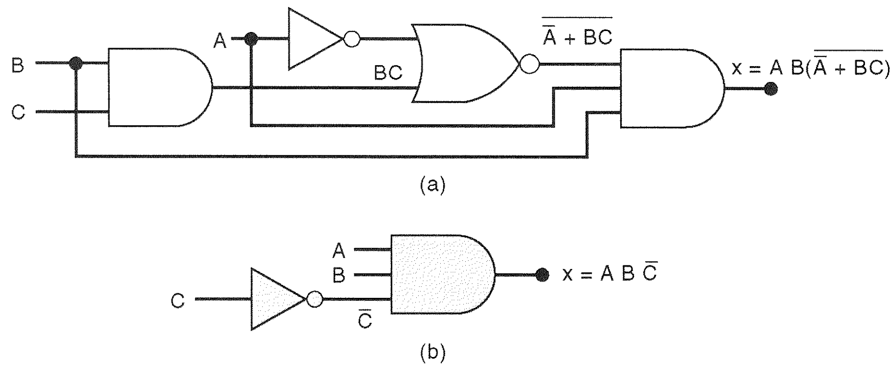


Fig. 4-1 Geralmente é possível simplificar um circuito lógico, como o que aparece em (a), e produzir uma implementação mais eficiente, mostrada em (b).

(c) $(A + B)(C + D + F)$
(d) $\overline{MN} + PQ$

2. Repita a Questão 1 para produto-de-somas.

4-2 SIMPLIFICAÇÃO DE CIRCUITOS LÓGICOS

Uma vez obtida a expressão de um circuito lógico, podemos ser capazes de reduzi-la a uma forma mais simples, que contenha um menor número de termos ou variáveis em um ou mais termos da expressão. Esta nova expressão pode ser usada para implementar um circuito que é equivalente ao circuito original, mas que contém um menor número de portas e conexões.

Para exemplificar, o circuito da Fig. 4-1(a) pode ser simplificado para produzir o circuito da Fig. 4-1(b). Uma vez que os circuitos implementam a mesma lógica, é óbvio que um circuito mais simples é mais desejável porque contém um menor número de portas e portanto será menor e mais barato do que o circuito original. Além disso, a confiabilidade será melhorada porque existe um menor número de ligações, diminuindo assim uma das causas potenciais de falhas no circuito.

Nas seções subseqüentes, estudaremos dois métodos utilizados para simplificar circuitos lógicos. Um dos métodos faz uso dos teoremas da álgebra booleana e, como veremos, é bastante dependente da inspiração e da experiência. O outro método (o mapa de Karnaugh) tem uma abordagem mais sistemática, com instruções passo a passo. Alguns professores podem querer omitir este método porque ele é bastante mecânico e provavelmente não contribui para uma melhor compreensão da álgebra booleana. Isto pode ser feito sem alterar a continuidade ou clareza do restante do texto.

4-3 SIMPLIFICAÇÃO ALGÉBRICA

Podemos usar os teoremas da álgebra booleana, que estudamos no Cap. 3, para nos ajudar a simplificar expressões para um circuito lógico. Infelizmente, nem sempre é óbvio qual teorema deve ser aplicado de modo a produzir o resultado mais simples. Além disso, não existe um modo fácil de constatar se a expressão obtida está em sua forma mais simples ou se poderia ser ainda mais simplificada. Portanto, a simplificação algébrica freqüentemente se torna um processo de tentativa e erro. Com a experiência, no entanto, pode-se ficar perito e obter resultados razoavelmente bons.

Os exemplos que se seguem ilustram muitas maneiras pelas quais os teoremas booleanos podem ser aplicados na tentativa de simplificar uma expressão. Você deve notar que estes exemplos contêm dois passos essenciais:

1. A expressão original é colocada sob a forma de soma-de-produtos pela aplicação repetitiva dos teoremas de DeMorgan e pela multiplicação de termos.
2. Uma vez que a expressão original esteja nesta forma, os termos produto são verificados quanto a fatores comuns, realizando-se a fatoração sempre que possível. Com sorte, a fatoração resulta na eliminação de um ou mais termos.

EXEMPLO 4-1

Simplifique o circuito lógico mostrado na Fig. 4-2(a).

Solução

O primeiro passo é determinar a expressão para a saída usando o método apresentado na Seção 3-6. O resultado é

$$z = ABC + A\overline{B} \cdot (\overline{\overline{A}\,\overline{C}})$$

Já com a expressão determinada, usualmente é uma boa idéia quebrar todos os grandes sinais de inversão usando os teoremas de DeMorgan e então multiplicar todos os termos.

$$\begin{aligned} z &= ABC + A\overline{B}(\overline{\overline{A}} + \overline{\overline{C}}) && \text{[teorema (17)]} \\ &= ABC + A\overline{B}(A + C) && \text{[cancela inversões duplas]} \\ &= ABC + A\overline{B}A + A\overline{B}C && \text{[multiplica]} \\ &= ABC + A\overline{B} + A\overline{B}C && [A \cdot A = A] \end{aligned}$$

Com a expressão agora sob a forma de soma-de-produtos, devemos procurar por variáveis comuns dentre os vários termos com a intenção de fatorar. O primeiro e terceiro termos têm AC em comum, que pode ser fatorado:

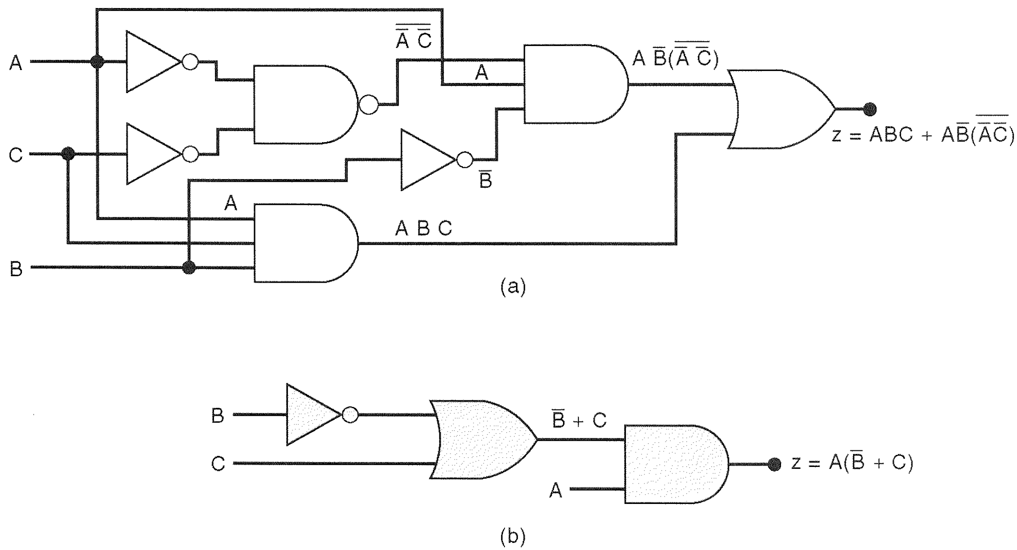


Fig. 4-2 Exemplo 4-1.

$$z = AC(B + \overline{B}) + A\overline{B}$$

Já que $B + \overline{B} = 1$, então

$$z = AC(1) + A\overline{B}$$
$$= AC + A\overline{B}$$

Podemos agora fatorar A, o que resulta em

$$z = A(C + \overline{B})$$

Este resultado não pode mais ser simplificado. A implementação do circuito é mostrada na Fig. 4-2(b). É óbvio que o circuito em (b) é bem mais simples do que o circuito original em (a).

EXEMPLO 4-2

Simplifique a expressão $z = ABC + AB\overline{C} + A\overline{B}C$.

Solução

Vamos ver dois modos diferentes de chegar ao mesmo resultado.

Método 1: Os primeiros dois termos na expressão têm o produto AB em comum. Logo,

$$z = AB(C + \overline{C}) + A\overline{B}C$$
$$= AB(1) + A\overline{B}C$$
$$= AB + A\overline{B}C$$

Podemos fatorar a variável A de ambos os termos:

$$z = A(B + \overline{B}C)$$

Aplicando o teorema (15),

$$z = A(B + C)$$

Método 2: A expressão original é $z = ABC + AB\overline{C} + A\overline{B}C$. Os primeiros dois termos têm AB em comum. O primeiro e o último termo têm AC em comum. Como saber se devemos fatorar AB dos primeiros dois termos ou AC dos dois termos extremos? Na verdade, podemos fazer ambos usando o termo ABC *duas vezes*. Em outras palavras, podemos reescrever a expressão como

$$z = ABC + AB\overline{C} + A\overline{B}C + ABC$$

onde somamos um termo extra ABC. Isto é válido e não altera o valor da expressão, tendo em vista que $ABC + ABC = ABC$ [teorema (7)]. Agora podemos fatorar AB dos dois primeiros termos e AC dos dois últimos termos:

$$z = AB(C + \overline{C}) + AC(\overline{B} + B)$$
$$= AB \cdot 1 + AC \cdot 1$$
$$= AB + AC = A(B + C)$$

Este é, naturalmente, o mesmo resultado obtido com o método 1. Esse artifício de usar o mesmo termo duas vezes sempre pode ser usado. De fato, o mesmo termo pode ser usado mais de duas vezes se for necessário.

EXEMPLO 4-3

Simplifique $z = \overline{A}C(\overline{\overline{A}BD}) + \overline{A}B\overline{C}\,\overline{D} + A\overline{B}C$.

Solução

Inicialmente, use o teorema de DeMorgan no primeiro termo:

$$z = \overline{A}C(A + \overline{B} + \overline{D}) + \overline{A}B\overline{C}\,\overline{D} + A\overline{B}C \quad \text{(passo 1)}$$

Multiplicando-se obtemos

$$z = \overline{A}CA + \overline{A}C\overline{B} + \overline{A}C\overline{D} + \overline{A}B\overline{C}\,\overline{D} + A\overline{B}C \quad (2)$$

Visto que $\overline{A} \cdot A = 0$, o primeiro termo é eliminado:

$$z = \overline{A}\,\overline{B}C + \overline{A}C\overline{D} + \overline{A}B\overline{C}\,\overline{D} + A\overline{B}C \quad (3)$$

Esta é a forma de soma-de-produtos desejada. Agora devemos procurar por fatores comuns dentre os vários termos produto. A idéia é investigar o maior fator comum entre quaisquer dois ou mais termos produto. Por exemplo, o primeiro e o último termo têm o fator comum $\overline{B}C$, e o segundo e o terceiro termo compartilham o fator comum $\overline{A}\,\overline{D}$. Podemos fatorá-los como se segue:

$$z = \overline{B}C(\overline{A} + A) + \overline{A}\,\overline{D}(C + B\overline{C}) \qquad (4)$$

Agora, sabendo que $\overline{A} + A = 1$, e $C + B\overline{C} = C + B$ [teorema (15)], temos

$$z = \overline{B}C + \overline{A}\,\overline{D}(B + C) \qquad (5)$$

Este mesmo resultado teria sido alcançado com outras escolhas para fatoração. Por exemplo, poderíamos ter fatorado C do primeiro, segundo e quarto termos produto, no passo 3, para obter

$$z = C(\overline{A}\,\overline{B} + \overline{A}\,\overline{D} + A\overline{B}) + \overline{A}B\overline{C}\,\overline{D}$$

A expressão entre parênteses pode ser fatorada ainda mais:

$$z = C(\overline{B}[\overline{A} + A] + \overline{A}\,\overline{D}) + \overline{A}B\overline{C}\,\overline{D}$$

Visto que $\overline{A} + A = 1$, ela se torna

$$z = C(\overline{B} + \overline{A}\,\overline{D}) + \overline{A}B\overline{C}\,\overline{D}$$

Multiplicando obtém-se

$$z = \overline{B}C + \overline{A}C\overline{D} + \overline{A}B\overline{C}\,\overline{D}$$

Agora podemos fatorar $\overline{A}\,\overline{D}$ do segundo e terceiro termos para obter

$$z = \overline{B}C + \overline{A}\,\overline{D}(C + B\overline{C})$$

Usando o teorema (15), a expressão entre parênteses se torna $B + C$. Assim, finalmente temos

$$z = \overline{B}C + \overline{A}\,\overline{D}(B + C)$$

Este é o mesmo resultado que obtivemos antes, mas exigiu muito mais passos. Isto ilustra por que devemos procurar pelos maiores fatores comuns: geralmente levará até a expressão final em menos passos.

EXEMPLO 4-4

Simplifique a expressão $x = (\overline{A} + B)(A + B + D)\overline{D}$.

Solução

A expressão pode ser colocada sob a forma de soma-de-produtos multiplicando-se todos os termos. O resultado é

$$x = \overline{A}A\overline{D} + \overline{A}B\overline{D} + \overline{A}D\overline{D} + BA\overline{D} + BB\overline{D} + BD\overline{D}$$

O primeiro termo pode ser eliminado, já que $\overline{A}A = 0$. Do mesmo modo, o terceiro e o sexto termo podem ser eliminados, visto que $D\overline{D} = 0$. O quinto termo pode ser simplificado para $B\overline{D}$, já que $BB = B$. Isto resulta em

$$x = \overline{A}B\overline{D} + AB\overline{D} + B\overline{D}$$

Podemos fatorar $B\overline{D}$ de cada termo para obter

$$x = B\overline{D}(\overline{A} + A + 1)$$

É claro que o termo entre parênteses é sempre 1, portanto, finalmente temos

$$x = B\overline{D}$$

EXEMPLO 4-5

Simplifique o circuito da Fig. 4-3(a).

Solução

A expressão para a saída z é

$$z = (\overline{A} + B)(A + \overline{B})$$

Multiplicando para conseguir a forma de soma-de-produtos, obtemos

$$z = \overline{A}A + \overline{A}\,\overline{B} + BA + B\overline{B}$$

Podemos eliminar $\overline{A}A = 0$ e $B\overline{B} = 0$ para terminar com

$$z = \overline{A}\,\overline{B} + AB$$

Esta expressão está implementada na Fig. 4-3(b), e se for comparada com o circuito original vemos que ambos os circuitos contêm o mesmo número de portas e conexões. Neste caso o processo de simplificação produziu um circuito equivalente, mas não um circuito mais simples.

EXEMPLO 4-6

Simplifique $x = A\overline{B}C + \overline{A}BD + \overline{C}\,\overline{D}$.

Solução

Você pode tentar, mas não será capaz de simplificar ainda mais esta expressão.

Questões de Revisão

1. Indique quais das seguintes expressões *não* estão sob a forma de soma-de-produtos:
   - **(a)** $RS\overline{T} + \overline{R}S\overline{T} + \overline{T}$
   - **(b)** $A\overline{DC} + \overline{A}DC$
   - **(c)** $MN\overline{P} + (M + \overline{N})P$
   - **(d)** $AB + \overline{A}B\overline{C} + A\overline{B}\,\overline{C}D$
2. Simplifique o circuito na Fig. 4-1(a) para chegar ao circuito da Fig. 4-1(b).
3. Troque cada porta AND na Fig. 4-1(a) por uma porta NAND. Determine a nova expressão de x e simplifique-a.

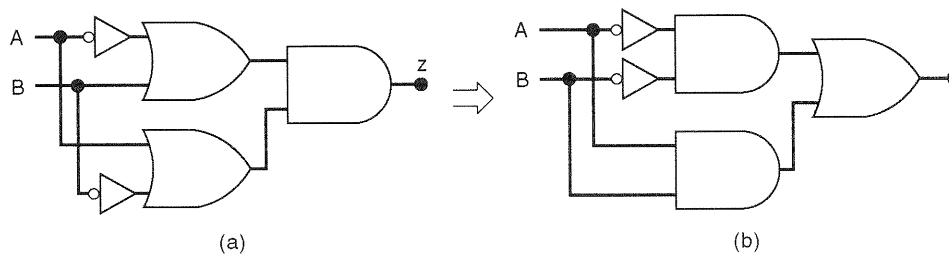
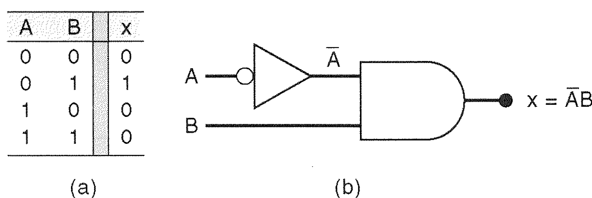


Fig. 4-3 Exemplo 4-5.

4-4 PROJETANDO CIRCUITOS LÓGICOS COMBINACIONAIS

Quando o nível de saída desejado de um circuito lógico é dado para todas as condições de entrada possíveis, os resultados podem ser convenientemente apresentados em uma tabela-verdade. A expressão booleana para o circuito pode ser derivada da tabela-verdade. Por exemplo, considere a Fig. 4-4(a), onde uma tabela-verdade é mostrada para um circuito que tem duas entradas, *A* e *B*, e uma saída *x*. A tabela mostra que a saída *x* está no nível 1 *somente* para o caso em que $A = 0$ e $B = 1$. Agora, resta determinar que circuito lógico produz esta operação. Deveria estar claro que uma solução possível é apresentada na Fig. 4-4(b). Nela, uma porta AND é usada com entradas $\overline{A}$ e *B*, de modo que $x = \overline{A} \cdot B$. Obviamente, *x* será 1 *somente se* ambas as entradas da porta AND forem 1, isto é, $\overline{A} = 1$ (o que significa $A = 0$) e $B = 1$. Para todos os outros valores de *A* e *B*, a saída *x* deve ser 0.



A	B	x
0	0	0
0	1	1
1	0	0
1	1	0

Fig. 4-4 Circuito que produz nível 1 na saída somente para a condição $A = 0$, $B = 1$.

Uma abordagem similar pode ser usada para outras condições de entrada. Por exemplo, se *x* tivesse que estar em alto somente para a condição $A = 1$, $B = 0$, o circuito resultante deveria ser uma porta AND com entradas *A* e $\overline{B}$. Em outras palavras, para qualquer uma das quatro possíveis condições de entrada podemos gerar uma saída alta *x* utilizando uma porta AND, com entradas apropriadas, para gerar o produto AND requerido. Os quatro casos distintos são mostrados na Fig. 4-5. Cada porta AND gera uma saída que é 1 *somente* para uma certa condição de entrada, e a saída é 0 para todas as outras condições. Deve-se notar que as entradas da AND são invertidas ou não, dependendo dos valores que as variáveis têm para a condição dada. Se a variável é 0 para a condição dada, ela é invertida antes de entrar na porta AND.

Vamos agora considerar o caso mostrado na Fig. 4-6(a), onde temos uma tabela-verdade indicando que a saída *x* deve ser 1 para dois casos distintos: $A = 0$, $B = 1$ e $A = 1$, $B = 0$. Como isto pode ser implementado? Sabemos que o termo AND $\overline{A} \cdot B$ gera 1 somente para a condição $A = 0$, $B = 1$, e o termo AND $A \cdot \overline{B}$ gera 1 para a condição $A = 1$, $B = 0$. Como *x* deve ser ALTO para uma *ou* outra condição, deve ficar claro que estes termos devem ser unidos com OR para produzirem a saída desejada *x*. Esta implementação é

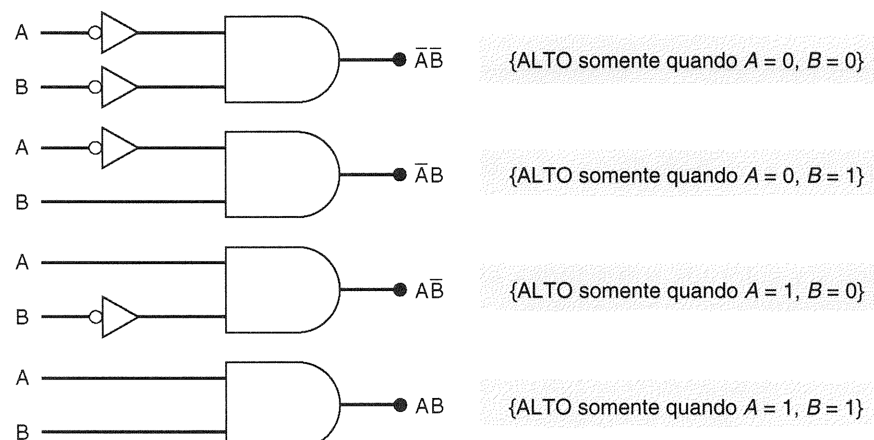


Fig. 4-5 Uma porta AND, com entradas apropriadas, pode ser usada para produzir uma saída em 1 para um conjunto específico de níveis de entrada.

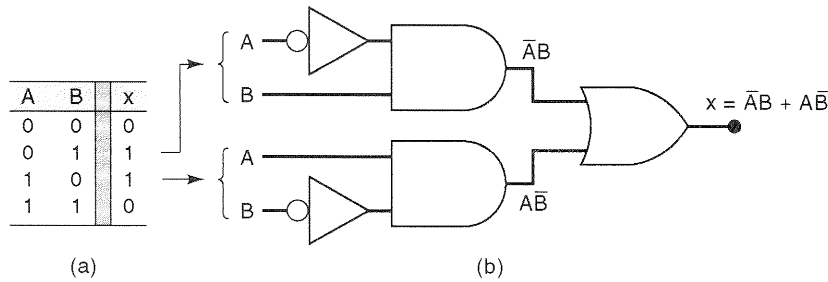


Fig. 4-6 Cada conjunto de condições de entrada que produz uma saída em ALTO é implementado por uma porta AND em separado. As saídas das portas AND são unidas com OR para produzir a saída final.

mostrada na Fig. 4-6(b), onde a expressão resultante para a saída é $x = \overline{A}B + A\overline{B}$.

Neste exemplo, um termo AND é gerado para cada caso na tabela onde a saída deve ser 1. As saídas das portas AND são então unidas com OR para produzir a saída x, que será 1 quando um dos termos AND for 1. Este mesmo procedimento pode ser estendido para exemplos com mais de duas entradas. Considere a tabela-verdade para um circuito de três entradas (Tabela 4-1). Nela existem três casos onde a saída x deve ser 1. O termo AND para cada caso está indicado. Novamente, observe que para cada caso onde a variável é 0 ela aparece complementada no termo AND. A expressão de soma-de-produtos para x é obtida unindo com OR os três termos AND.

$$x = \overline{A}B\overline{C} + \overline{A}BC + ABC$$

TABELA 4-1

A	B	C	x	
0	0	0	0	
0	0	1	0	
0	1	0	1	$\rightarrow \overline{A}B\overline{C}$
0	1	1	1	$\rightarrow \overline{A}BC$
1	0	0	0	
1	0	1	0	
1	1	0	0	
1	1	1	1	$\rightarrow ABC$

Procedimento Completo de Projeto

Uma vez que a expressão da saída tenha sido determinada da tabela-verdade sob a forma de soma-de-produtos, ela pode ser facilmente implementada usando portas AND, OR e INVERSORes. Usualmente, entretanto, a expressão pode ser simplificada, resultando num circuito mais eficiente. O exemplo seguinte ilustra o procedimento completo de projeto.

EXEMPLO 4-7

Projete um circuito lógico que tem três entradas, A, B e C, e cuja saída vai para ALTO somente quando a maioria das entradas está em ALTO.

Solução

Passo 1. Monte a tabela-verdade.

Com base no enunciado do problema, a saída x deve ser 1 sempre que duas ou mais entradas forem 1. Para todos os outros casos, a saída deve ser 0 (Tabela 4-2).

TABELA 4-2

A	B	C	x	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\rightarrow \overline{A}BC$
1	0	0	0	
1	0	1	1	$\rightarrow A\overline{B}C$
1	1	0	1	$\rightarrow AB\overline{C}$
1	1	1	1	$\rightarrow ABC$

Passo 2. Escreva o termo AND para cada caso onde a saída é 1.

Existem quatro destes casos. Os termos AND estão mostrados próximos à tabela (Tabela 4-2). Note mais uma vez que cada termo AND contém cada variável de entrada, invertida ou não.

Passo 3. Escreva a expressão da soma-de-produtos para a saída.

$$x = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

Passo 4. Simplifique a expressão de saída.

Esta expressão pode ser simplificada de muitos modos. Talvez o modo mais rápido seja reparar que o último termo ABC tem duas variáveis em comum com cada um dos outros termos. Logo, podemos usar o termo ABC para fatorar com cada um dos outros. A expressão é reescrita com o

termo ABC aparecendo três vezes (lembre-se do Exemplo 4-2 que isto é permitido em álgebra booleana):

$$x = \overline{A}BC + ABC + A\overline{B}C + ABC + AB\overline{C} + ABC$$

Fatorando os pares de termos apropriados, temos

$$x = BC(\overline{A} + A) + AC(\overline{B} + B) + AB(\overline{C} + C)$$

Visto que cada termo entre parênteses é igual a 1, temos

$$x = BC + AC + AB$$

Passo 5. Implemente o circuito para a expressão final.

Esta expressão está implementada na Fig. 4-7. Como a expressão está sob a forma de soma-de-produtos, o circuito consiste em um grupo de portas AND ligadas em uma única porta OR.

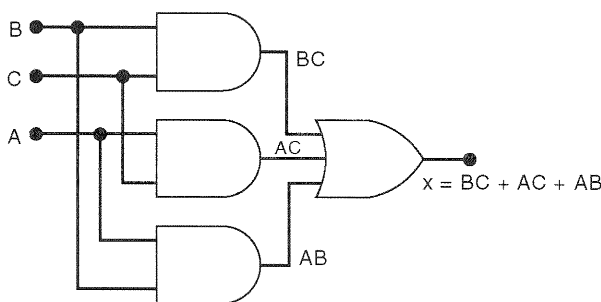


Fig. 4-7 Exemplo 4-7.

EXEMPLO 4-8

Veja a Fig. 4-8(a), onde um conversor analógico-digital está monitorando a tensão de uma bateria de 12 V de uma espaçonave em órbita. A saída do conversor é um número binário de quatro bits, $ABCD$, que corresponde à tensão da bateria em degraus de 1 V, sendo A o MSB. As saídas binárias do conversor são ligadas em um circuito digital que deve produzir uma saída em ALTO sempre que o valor binário for maior do que $0110_2 = 6_{10}$, ou seja, quando a tensão da bateria for maior do que 6 V. Projete este circuito lógico.

Solução

A tabela-verdade é mostrada na Fig. 4-8(b). Para cada linha da tabela-verdade indicamos o equivalente decimal do número representado pela combinação $ABCD$.

A saída z é igual a 1 para todos os casos onde o número binário é maior do que 0110. Para todos os outros casos, z é igual a 0. Esta tabela-verdade fornece a seguinte expressão de soma-de-produtos:

$$z = \overline{A}BCD + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D + A\overline{B}C\overline{D} + A\overline{B}CD + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABC\overline{D} + ABCD$$

Simplificar esta expressão é uma tarefa tremenda, mas com um pouco de cuidado ela pode ser feita. O processo passo a passo envolve fatorar e eliminar termos da forma $A + \overline{A}$:

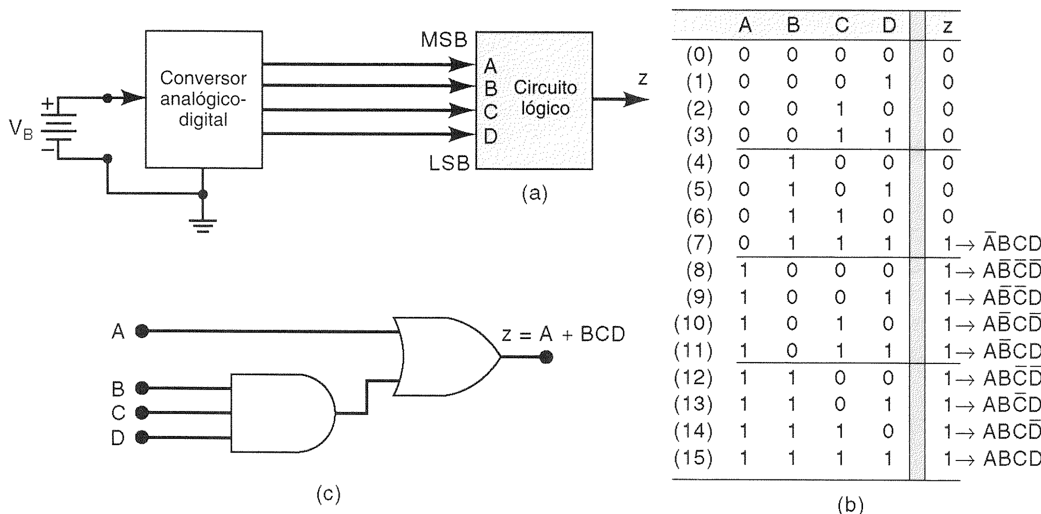
$$\begin{aligned} z &= \overline{A}BCD + A\overline{B}\,\overline{C}(\overline{D} + D) + A\overline{B}C(\overline{D} + D) + AB\overline{C}(\overline{D} + D) + ABC(\overline{D} + D) \\ &= \overline{A}BCD + A\overline{B}\,\overline{C} + A\overline{B}C + AB\overline{C} + ABC \\ &= \overline{A}BCD + A\overline{B}(\overline{C} + C) + AB(\overline{C} + C) \\ &= \overline{A}BCD + A\overline{B} + AB \\ &= \overline{A}BCD + A(\overline{B} + B) \\ &= \overline{A}BCD + A \end{aligned}$$

Isto pode ainda ser reduzido aplicando-se o teorema (15), que é: $x + \overline{x}y = x + y$. Neste caso $x = A$ e $y = BCD$. Logo,

$$z = \overline{A}BCD + A = BCD + A$$

Esta expressão final está implementada na Fig. 4-8(c).

Como esse exemplo demonstra, o método da simplificação algébrica pode ser maçante quando a expressão original contém um grande número de termos. Esta é uma limitação que não é partilhada pelo método do mapa de Karnaugh, como veremos posteriormente.



	A	B	C	D	z
(0)	0	0	0	0	0
(1)	0	0	0	1	0
(2)	0	0	1	0	0
(3)	0	0	1	1	0
(4)	0	1	0	0	0
(5)	0	1	0	1	0
(6)	0	1	1	0	0
(7)	0	1	1	1	1 → $\overline{A}BCD$
(8)	1	0	0	0	1 → $A\overline{B}\,\overline{C}\,\overline{D}$
(9)	1	0	0	1	1 → $A\overline{B}\,\overline{C}D$
(10)	1	0	1	0	1 → $A\overline{B}C\overline{D}$
(11)	1	0	1	1	1 → $A\overline{B}CD$
(12)	1	1	0	0	1 → $AB\overline{C}\,\overline{D}$
(13)	1	1	0	1	1 → $AB\overline{C}D$
(14)	1	1	1	0	1 → $ABC\overline{D}$
(15)	1	1	1	1	1 → $ABCD$

(b)

Fig. 4-8 Exemplo 4-8.

Implementando o Projeto Final

Nos exemplos de projetos apresentados, o circuito final foi implementado usando-se portas AND e OR. De fato, a forma de soma-de-produtos sempre produz um circuito que contém uma ou mais portas AND acionando uma porta OR. Uma das razões para a utilização da forma de soma-de-produtos é que ela pode ser implementada usando-se apenas portas NAND com pouco, ou nenhum, aumento de complexidade em relação à implementação AND/OR. Tendo em vista que as portas NAND são as portas lógicas mais disponíveis na família lógica TTL, esta é uma característica importante.

Para ilustrar, a Fig. 4-9 mostra as implementações equivalentes com portas NAND para os circuitos das Figs. 4-7 e 4-8(c). Você pode fazer essas conversões como revisão do procedimento apresentado no Cap. 3.

Comparando a implementação NAND com o circuito original na Fig. 4-9(a), observa-se que eles são idênticos na estrutura, isto é, *cada* porta do circuito original foi substituída por uma única porta NAND. Esta característica é verdadeira somente se o circuito original está sob a forma de soma-de-produtos. A única exceção é quando a forma de soma-de-produtos contém um termo de uma variável tal como $z = A + BCD$ na Fig. 4-9(b). Neste caso a implementação NAND requer uma porta NAND extra usada como INVERSOR na entrada A.

Podemos resumir o processo de conversão de um circuito sob a forma de *soma-de-produtos* com portas AND/OR para portas NAND como se segue:

1. Substitua cada porta AND, porta OR e INVERSOR por uma *única* porta NAND.
2. Utilize uma porta NAND para inverter qualquer variável simples que aciona a porta OR final.

Verifique este processo para os circuitos na Fig. 4-9.

EXEMPLO 4-9

Veja a Fig. 4-10(a). Numa máquina copiadora simples, um sinal de parada, S, deve ser gerado para interromper a operação da máquina e energizar uma luz indicadora, sempre que uma das seguintes condições existir: (1) a bandeja de alimentação de papel estiver vazia; ou (2) as duas chaves na trajetória do papel estiverem ativadas, indicando um congestionamento no caminho do papel. A presença de papel na bandeja de alimentação é indicada por um sinal lógico P em ALTO. Cada chave produz um sinal lógico (Q e R) que vai para ALTO sempre que o papel passa sobre a chave para ativá-la. Projete o circuito lógico para produzir um nível ALTO no sinal de saída S para as condições estabelecidas, e implemente-o usando o chip 74LS00 (Fig. 3-31).

Solução

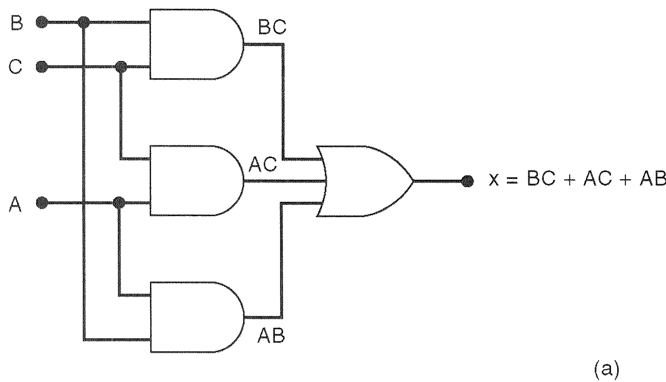
Utilizaremos o processo de cinco passos usado no Exemplo 4-7.

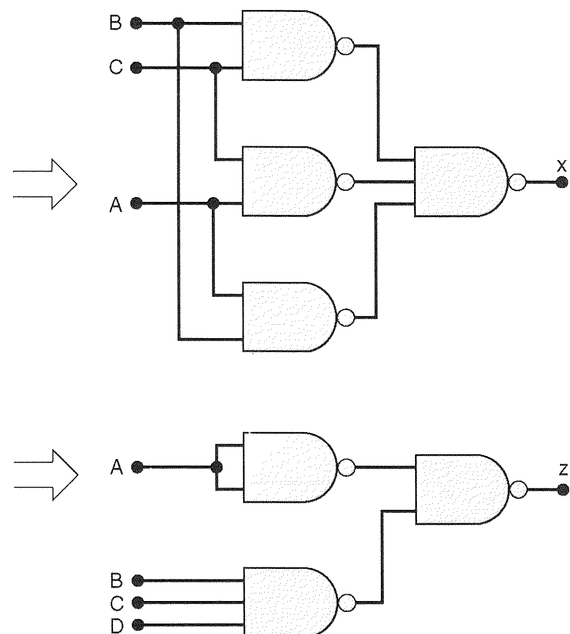
A tabela-verdade está na Tabela 4-3. A saída S assume o valor lógico 1 sempre que $P = 0$, já que isto indica que não há papel na bandeja de alimentação. S também é 1 para os dois casos em que Q e R são ambos 1, indicando um congestionamento de papel. Conforme a tabela mostra, existem cinco diferentes condições de entrada que produzem uma saída em ALTO. (Passo 1)

Os termos AND para cada um dos casos estão indicados. (Passo 2)

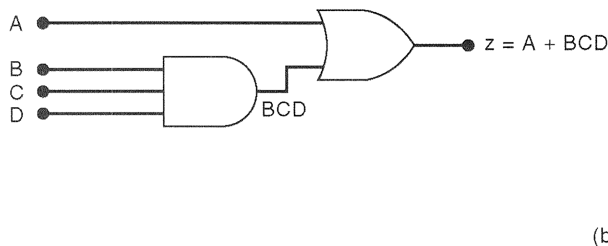
A expressão da soma-de-produtos se torna

$$S = \overline{P}\,\overline{Q}\,\overline{R} + \overline{P}\,\overline{Q}R + \overline{P}Q\overline{R} + \overline{P}QR + PQR \quad \text{(Passo 3)}$$





(a)



(b)

Fig. 4-9 (a) Conversão do circuito da Fig. 4-7 para NANDs; (b) conversão do circuito da Fig. 4-8(c) para NANDs.

Podemos começar a simplificação fatorando $\overline{P}\overline{Q}$ dos termos 1 e 2 e fatorando $\overline{P}Q$ dos termos 3 e 4:

$$S = \overline{P}\,\overline{Q}(\overline{R} + R) + \overline{P}Q(\overline{R} + R) + PQR \qquad \text{(Passo 4)}$$

Agora podemos eliminar os termos $\overline{R} + R$, já que são iguais a 1:

$$S = \overline{P}\,\overline{Q} + \overline{P}Q + PQR$$

Fatorar $\overline{P}$ dos termos 1 e 2 permite a eliminação de Q destes termos:

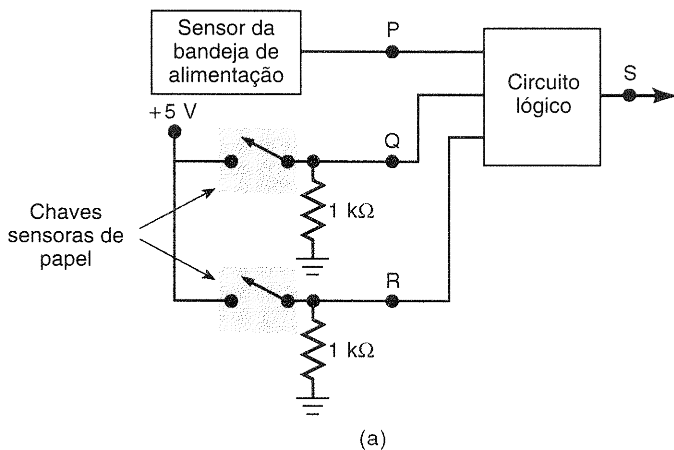
$$S = \overline{P} + PQR$$

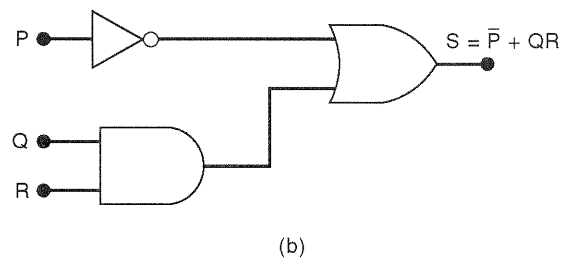
Aplicando o teorema (15) ($x + \overline{x}y = x + y$) obtemos

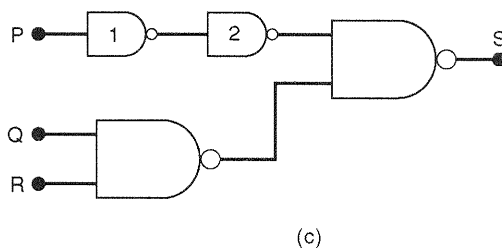
$$S = \overline{P} + QR$$

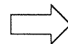
TABELA 4-3

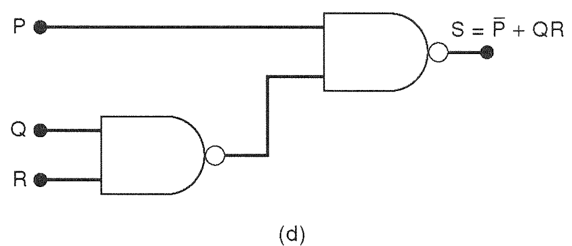
P	*Q*	*R*	*S*	
0	0	0	1	$\overline{P}\,\overline{Q}\,\overline{R}$
0	0	1	1	$\overline{P}\,\overline{Q}R$
0	1	0	1	$\overline{P}Q\overline{R}$
0	1	1	1	$\overline{P}QR$
1	0	0	0	
1	0	1	0	
1	1	0	0	
1	1	1	1	PQR











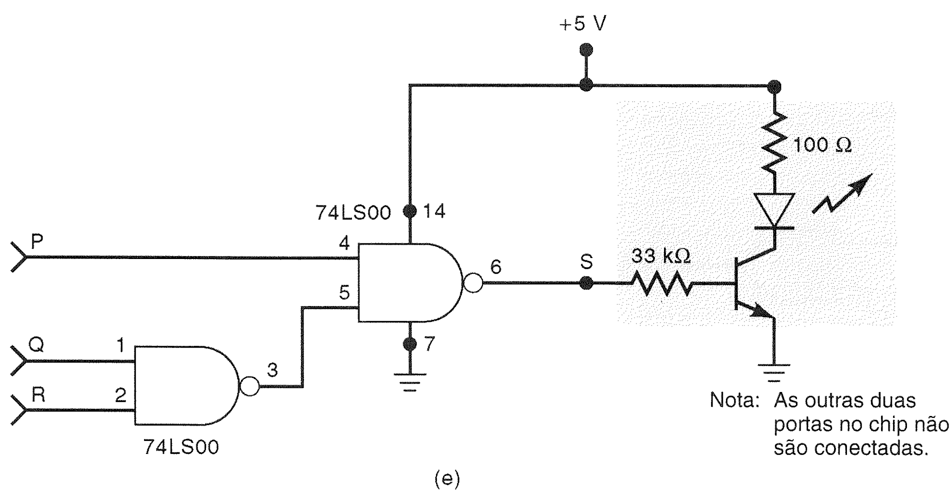


Fig. 4-10 Procedimento completo de projeto (Exemplo 4-9) implementado usando um chip NAND 74LS00.

A implementação AND/OR para este circuito está ilustrada na Fig. 4-10(b). (Passo 5)

Tendo em vista que o circuito deve ser implementado com o chip 74LS00, que tem quatro portas NAND de duas entradas, o circuito da Fig. 4-10(b) deve ser convertido para utilizar apenas portas NAND. Substitui-se cada porta OR e AND por uma porta NAND e troca-se o INVERSOR pela porta NAND INVERSORa [identificada com 1 na Fig. 4-10(c)]. Além disso, como a entrada superior da porta OR é uma variável simples (*P*), uma porta NAND INVERSORa (identificada com 2) deve ser colocada nessa entrada. Obviamente, os dois INVERSORes podem ser eliminados para obter o circuito com NANDs da Fig. 4-10(d).

A Fig. 4-10(e) é a versão final do circuito mostrando a pinagem do CI, incluindo os pinos de alimentação (+5 V e TERRA) e o transistor de acionamento de saída com o LED indicador para o sinal *S*.

Questões de Revisão

1. Escreva a expressão de soma-de-produtos para um circuito com quatro entradas e uma saída que deve estar em ALTO somente quando a entrada *A* está em BAIXO, ao mesmo tempo em que exatamente duas outras entradas estão em BAIXO.
2. Implemente a expressão da Questão 1 usando apenas portas NAND de quatro entradas. Quantas são necessárias?

4-5 MÉTODO DO MAPA DE KARNAUGH

O **mapa de Karnaugh** é um método gráfico usado para simplificar uma equação lógica ou para converter uma tabela-verdade no seu circuito lógico correspondente, de um modo simples e ordenado. Embora um mapa de Karnaugh (daqui para a frente abreviado como **mapa K**) possa ser usado em problemas que envolvem qualquer número de variáveis de entrada, sua utilidade prática está limitada a seis variáveis. A apresentação que se segue está restrita a problemas com até quatro entradas, pois mesmo os problemas com cinco ou seis entradas são demasiadamente complicados, sendo mais bem resolvidos por um programa de computador.

Formato do Mapa de Karnaugh

O mapa K, como uma tabela-verdade, é um meio de mostrar a relação entre as entradas lógicas e a saída desejada. A Fig. 4-11 apresenta três exemplos de mapas K, para duas, para três e para quatro variáveis, em conjunto com as tabelas-verdade correspondentes. Estes exemplos ilustram os seguintes pontos importantes:

1. A tabela-verdade fornece o valor da saída *X* para cada combinação de valores da entrada. O mapa K fornece a mesma informação num formato diferente. Cada linha na tabela-verdade corresponde a um quadrado no mapa K. Por exemplo, na Fig. 4-11(a), a condição $A = 0$, $B = 0$, na tabela-verdade, corresponde ao quadrado $\overline{A}\,\overline{B}$ no mapa K. Como a tabela-verdade mostra $X = 1$ para este caso, 1 é colocado no quadrado $\overline{A}\,\overline{B}$ no mapa K. Do mesmo modo, a condição $A = 1$, $B = 1$ na tabela-verdade corresponde ao quadrado AB no mapa K. Como $X = 1$ para este caso, 1 é colocado no quadrado AB. Todos os outros quadrados são preenchidos com 0s. Esta mesma idéia é usada nos mapas de três e quatro variáveis mostrados na figura.
2. Os quadrados do mapa K são identificados de modo que quadrados adjacentes horizontalmente diferem apenas em um variável. Por exemplo, o quadrado do canto superior esquerdo no mapa de quatro variáveis é $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$, enquanto o quadrado imediatamente à sua direita é $\overline{A}\,\overline{B}\,\overline{C}D$ (apenas a variável *D* é diferente). Do mesmo modo, quadrados adjacentes verticalmente diferem apenas em uma variável. Por exemplo, o quadrado do canto superior esquerdo no mapa de quatro variáveis é $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$, enquanto o quadrado diretamente abaixo dele é $\overline{A}B\overline{C}\,\overline{D}$ (apenas a variável *B* é diferente).

   Note que cada quadrado na linha superior é considerado adjacente ao quadrado correspondente na linha inferior. Por exemplo, o quadrado $\overline{A}\,\overline{B}CD$ na linha superior é adjacente ao quadrado $A\overline{B}CD$ na linha inferior, pois diferem apenas na variável *A*. Você pode imaginar que a parte superior do mapa foi dobrada para tocar a parte inferior. Analogamente, os quadrados da coluna mais à esquerda são adjacentes aos quadrados correspondentes da coluna mais à direita.
3. Para que os quadrados adjacentes, tanto na horizontal quanto na vertical, difiram em apenas uma variável, a identificação de cima para baixo deve ser feita na ordem mostrada: $\overline{A}\,\overline{B}$, $\overline{A}B$, AB, $A\overline{B}$. O mesmo se aplica à identificação da esquerda para a direita.
4. Uma vez que um mapa K foi preenchido com 0s e 1s, a expressão da soma-de-produtos para a saída *X* pode ser obtida juntando-se com OR os quadrados que contêm 1. No mapa de três variáveis da Fig. 4-11(b), os quadrados $\overline{A}\,\overline{B}\,\overline{C}$, $\overline{A}\,\overline{B}C$, $\overline{A}B\overline{C}$ e $AB\overline{C}$ contêm 1, portanto $X = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + ABC$.

Agrupamento de Termos no Mapa

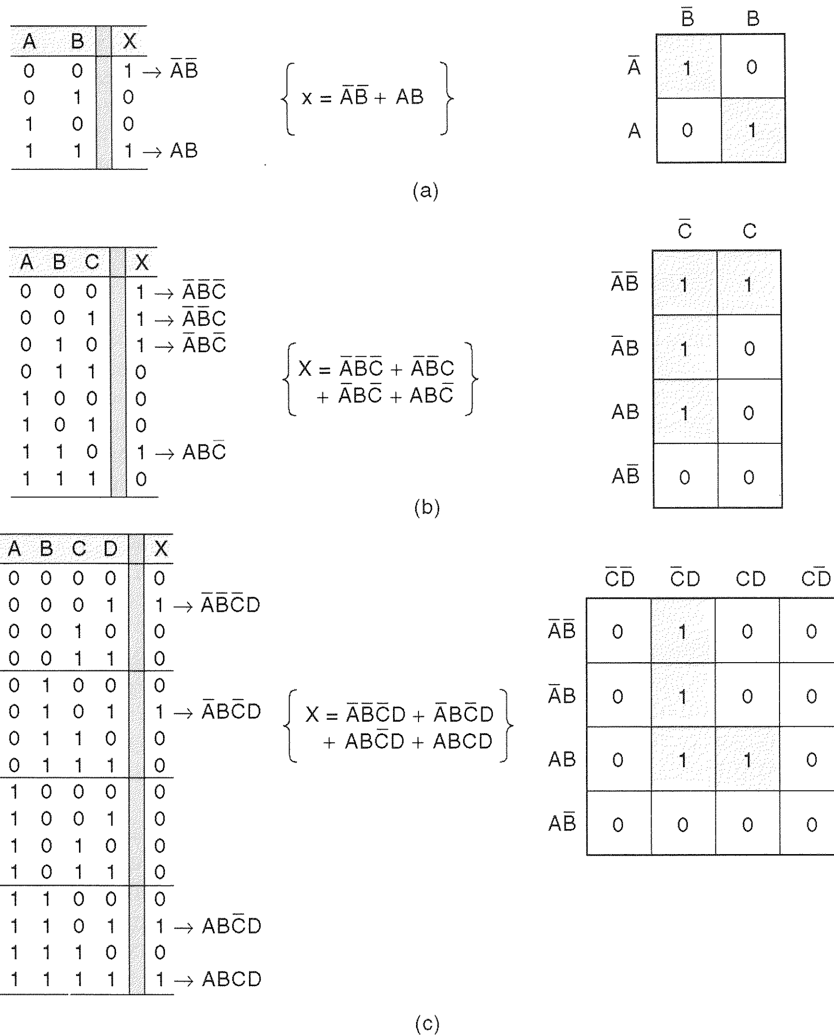
A expressão para a saída *X* pode ser simplificada combinando-se adequadamente os quadrados no mapa K que contêm 1. O processo de combinar estes 1s é chamado de **agrupamento**.

Agrupando Dois Termos (Pares)

Na Fig. 4-12(a) está o mapa K para uma determinada tabela-verdade de três variáveis. Este mapa contém um par de 1s que são adjacentes na vertical: o primeiro representa $\overline{A}B\overline{C}$ e o segundo representa $AB\overline{C}$. Repare que nestes dois termos apenas a variável *A* aparece tanto na forma normal quanto na complementar (*B* e $\overline{C}$ permanecem inalterados). Estes dois termos podem ser agrupados (combinados) para dar um resultado que elimina a variável *A*, visto que ela aparece em ambas as formas, normal e complementar. Isto é facilmente provado como se segue:

$$\begin{aligned} X &= \overline{A}B\overline{C} + AB\overline{C} \\ &= B\overline{C}(\overline{A} + A) \\ &= B\overline{C}(1) = B\overline{C} \end{aligned}$$

Este mesmo princípio permanece válido para qualquer par de 1s adjacentes na vertical ou na horizontal. A Fig. 4-12(b) mostra um exemplo de dois 1s horizontalmente adja-



A	B	X	
0	0	1	→ $\overline{A}\overline{B}$
0	1	0	
1	0	0	
1	1	1	→ AB

$$x = \overline{A}\overline{B} + AB$$

	$\overline{B}$	B
$\overline{A}$	1	0
A	0	1

(a)

A	B	C	X	
0	0	0	1	→ $\overline{A}\overline{B}\overline{C}$
0	0	1	1	→ $\overline{A}\overline{B}C$
0	1	0	1	→ $\overline{A}B\overline{C}$
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	1	→ $AB\overline{C}$
1	1	1	0	

$$X = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}B\overline{C} + AB\overline{C}$$

	$\overline{C}$	C
$\overline{A}\overline{B}$	1	1
$\overline{A}B$	1	0
AB	1	0
$A\overline{B}$	0	0

(b)

A	B	C	D	X	
0	0	0	0	0	
0	0	0	1	1	→ $\overline{A}\overline{B}\overline{C}D$
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	→ $\overline{A}B\overline{C}D$
0	1	1	0	0	
0	1	1	1	0	
1	0	0	0	0	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	1	→ $AB\overline{C}D$
1	1	1	0	0	
1	1	1	1	1	→ $ABCD$

$$X = \overline{A}\overline{B}\overline{C}D + \overline{A}B\overline{C}D + AB\overline{C}D + ABCD$$

	$\overline{C}\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\overline{B}$	0	1	0	0
$\overline{A}B$	0	1	0	0
AB	0	1	1	0
$A\overline{B}$	0	0	0	0

(c)

Fig. 4-11 Mapas de Karnaugh e tabelas-verdade para (a) duas, (b) três e (c) quatro variáveis.

centes. Estes dois podem ser agrupados e a variável C eliminada, já que ela aparece nas formas não-complementada e complementada para resultar em $X = \overline{A}B$.

Um outro exemplo está ilustrado na Fig. 4-12(c). Num mapa K a linha superior e a linha inferior são consideradas adjacentes. Assim, os dois 1s neste mapa podem ser agrupados para produzir como resultado $\overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} = \overline{B}\overline{C}$.

A Fig. 4-12(d) mostra um mapa K que tem dois pares de 1s que podem ser agrupados. Os dois 1s na linha superior são horizontalmente adjacentes. Os dois 1s na linha inferior também são adjacentes, já que, em um mapa K, a coluna de quadrados mais à esquerda é considerada adjacente com a coluna mais à direita. Quando o par de 1s superior é agrupado, a variável D é eliminada (pois ela aparece tanto como D quanto como $\overline{D}$) para gerar o termo $\overline{A}\overline{B}C$. Agrupar o par inferior elimina a variável C para gerar o termo $A\overline{B}\overline{D}$. Estes dois termos são unidos por um OR, obtendo-se o resultado final para X.

Resumindo:

Agrupar um par de 1s adjacentes num mapa K elimina a variável que aparece nas formas complementada e não-complementada.

Agrupando Quatro Termos (Quartetos)

Um mapa K pode conter um grupo de quatro 1s adjacentes entre si. Este grupo é denominado *quarteto*. A Fig. 4-13 mostra vários exemplos de quartetos. Na parte (a) os quatro 1s são verticalmente adjacentes, e na parte (b) eles são adjacentes na horizontal. O mapa K na Fig. 4-13(c) contém quatro 1s formando um quadrado, e eles são considerados adjacentes entre si. Os quatro 1s na Fig. 4-13(d) também são adjacentes, assim como os da Fig. 4-13(e) porque, conforme apresentado anteriormente, as linhas superior e inferior são consideradas adjacentes entre si, do mesmo modo que as colunas mais à esquerda e mais à direita.

Quando um quarteto é agrupado, o termo resultante contém apenas as variáveis que não mudam de forma para todos os quadrados do quarteto. Por exemplo, na Fig. 4-13(a), os quatro quadrados que contêm 1 são $\overline{A}\overline{B}C$, $\overline{A}BC$, ABC e $A\overline{B}C$. Um exame destes termos revela que apenas a variável C permanece inalterada (tanto A como B aparecem nas formas não-complementada e complementada). Assim, a expressão resultante para X é simplesmente $X = C$. Isto pode ser provado como se segue:

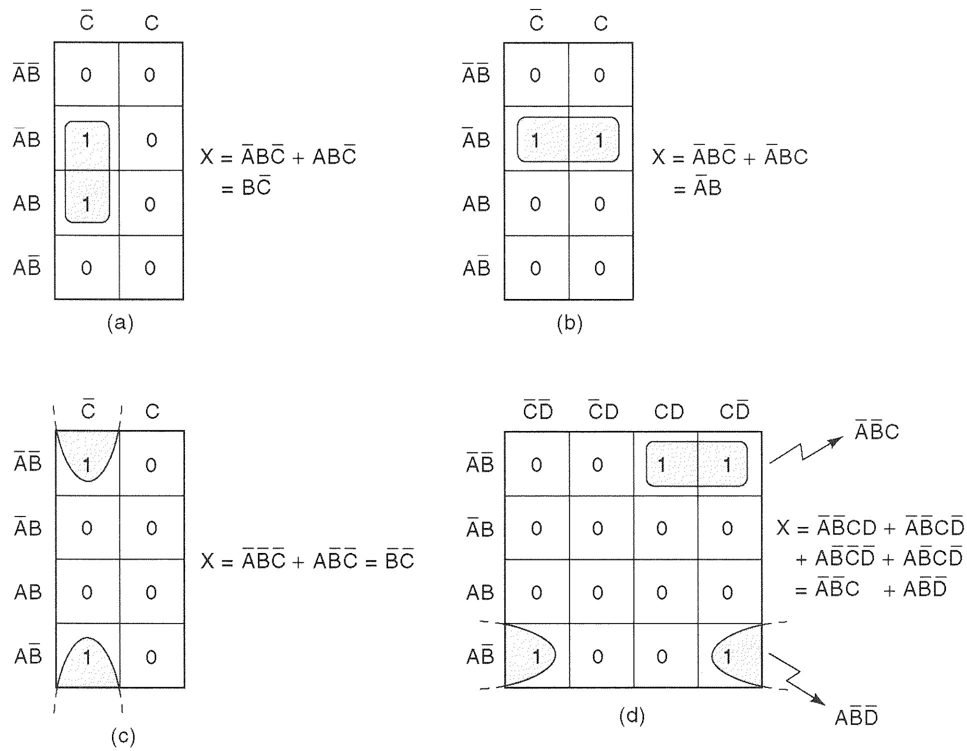


Fig. 4-12 Exemplos de agrupamentos de pares de 1s adjacentes.

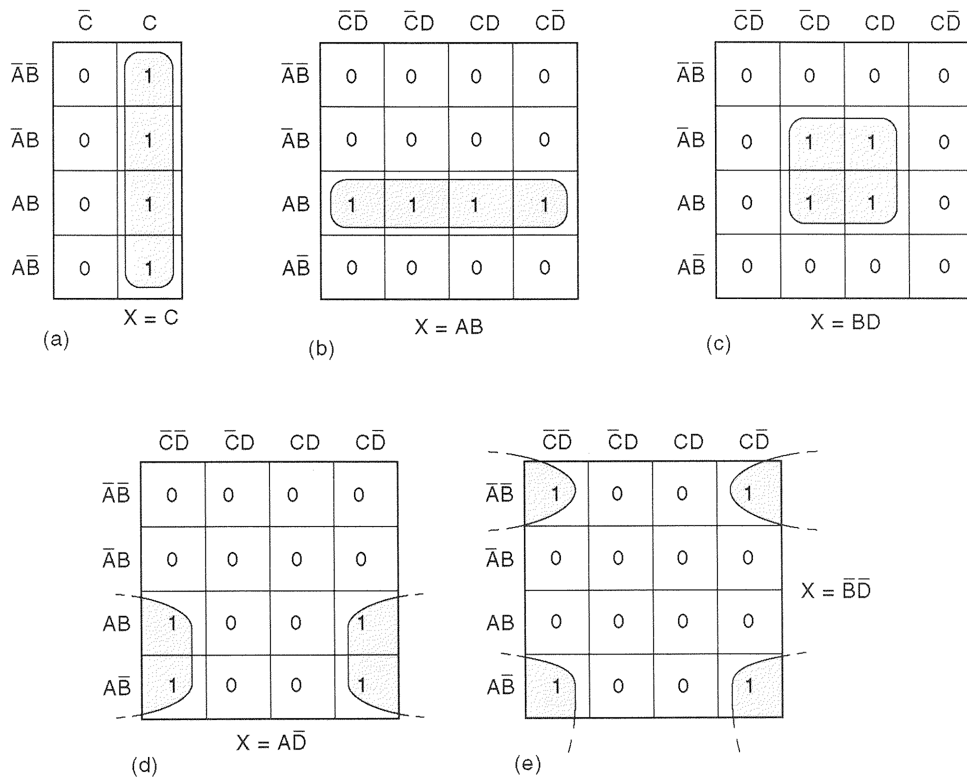


Fig. 4-13 Exemplos de agrupamentos de quatro 1s (quartetos).

$$
\begin{aligned}
X &= \overline{A}\overline{B}C + \overline{A}BC + ABC + A\overline{B}C \\
&= \overline{A}C(\overline{B} + B) + AC(B + \overline{B}) \\
&= \overline{A}C + AC \\
&= C(\overline{A} + A) = C
\end{aligned}
$$

Como outro exemplo, considere a Fig. 4-13(d), onde os quatro quadrados que contêm 1s são: $AB\overline{C}\overline{D}$, $A\overline{B}\overline{C}\overline{D}$, $ABC\overline{D}$ e $A\overline{B}C\overline{D}$. Um exame destes termos indica que somente as variáveis A e $\overline{D}$ permanecem inalteradas, portanto a expressão simplificada para X é

$$X = A\overline{D}$$

Isto pode ser provado da mesma maneira que foi feito anteriormente. O leitor deve analisar cada um dos casos na Fig. 4-13 para verificar as expressões indicadas para X.

Resumindo:

Agrupar um quarteto de 1s elimina as duas variáveis que aparecem nas formas complementada e não-complementada.

Agrupando Oito Termos (Octetos)

Um grupo de oito 1s que são adjacentes entre si é chamado de *octeto*. Muitos exemplos de octetos são mostrados na Fig. 4-14. Quando um octeto é agrupado num mapa de quatro variáveis, três das quatro variáveis são eliminadas, porque apenas uma variável permanece inalterada. Por exemplo, um exame dos oito quadrados agrupados na Fig. 4-14(a) mostra que somente a variável B está na mesma forma para todos os oito quadrados; as outras variáveis aparecem nas formas complementada e não-complementada. Portanto, para este mapa, $X = B$. O leitor pode verificar os resultados para os outros exemplos na Fig. 4-14.

Resumindo:

Agrupar um octeto de 1s elimina as três variáveis que aparecem nas formas complementada e não-complementada.

Processo Completo de Simplificação

Vimos que o agrupamento de pares, quartetos e octetos num mapa K pode ser usado para obtermos uma expressão simplificada. Podemos resumir a regra para grupos de *qualquer* tamanho:

Quando uma variável aparece nas formas complementada e não-complementada dentro de um grupo, esta variável é eliminada da expressão. Variáveis que não mudam para todos os quadrados do grupo devem aparecer na expressão final.

Deve ficar claro que um grupo maior de 1s elimina mais variáveis. Para ser exato, um grupo de dois elimina uma variável, um grupo de quatro elimina duas e um grupo de oito elimina três. Este princípio, agora, será utilizado para obter uma expressão lógica simplificada a partir de um mapa K que contenha qualquer combinação de 1s e 0s.

O procedimento será primeiramente resumido e então aplicado em vários exemplos. Os passos a seguir são reali-

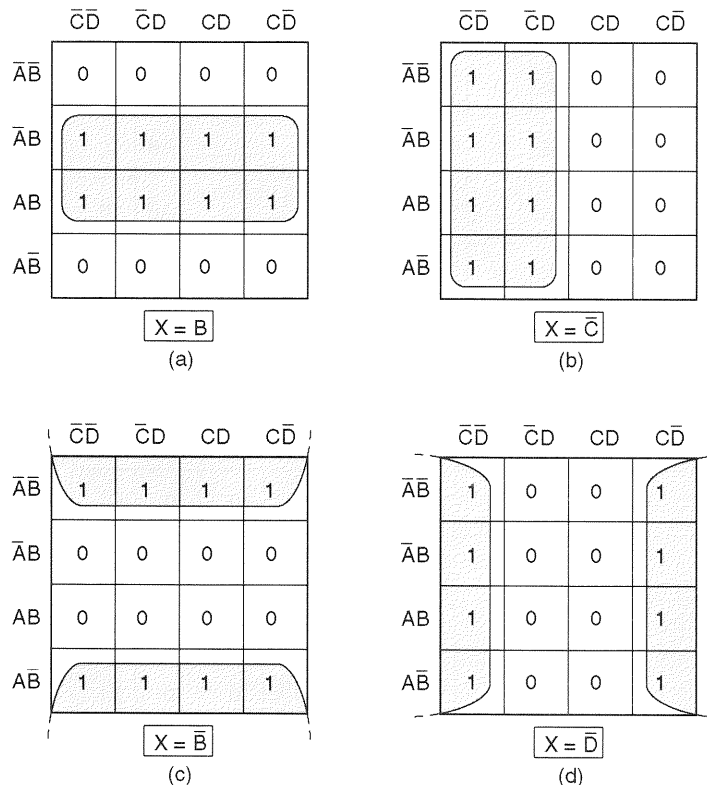


Fig. 4-14 Exemplos de agrupamentos de oito 1s (octetos).

zados para a utilização do método do mapa K para simplificação de uma expressão booleana:

Passo 1 Construa o mapa K e coloque 1s nos quadrados que correspondem aos 1s na tabela-verdade. Coloque 0s nos outros quadrados.

Passo 2 Examine o mapa para detectar 1s adjacentes e agrupe aqueles 1s que *não* são adjacentes a quaisquer outros 1s. Estes são denominados 1s *isolados*.

Passo 3 Em seguida, procure por aqueles 1s que são adjacentes a somente um outro 1. Agrupe *todo* par que contém tal 1.

Passo 4 Agrupe qualquer octeto, mesmo que ele contenha alguns 1s que já tenham sido combinados.

Passo 5 Agrupe qualquer quarteto que contém um ou mais 1s que ainda não tenham sido combinados, *certificando-se de usar o número mínimo de agrupamentos.*

Passo 6 Agrupe quaisquer pares necessários para incluir quaisquer 1s que ainda não tenham sido combinados, *certificando-se de usar o número mínimo de agrupamentos.*

Passo 7 Forme a soma OR de todos os termos gerados por cada agrupamento.

Estes passos são seguidos e mencionados nos exemplos seguintes. Em cada caso, a expressão lógica resultante está na sua forma de soma-de-produtos mais simples.

EXEMPLO 4-10

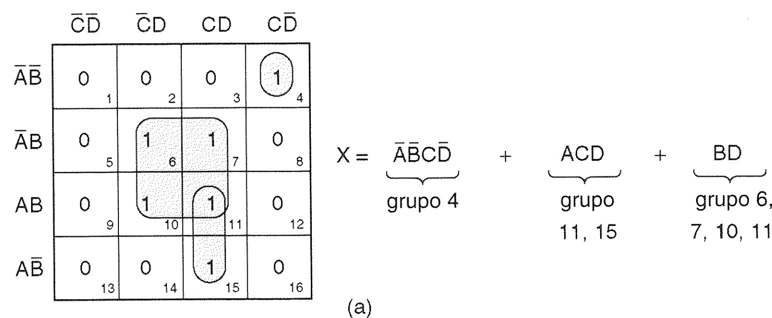
A Fig. 4-15(a) mostra o mapa K para um problema de quatro variáveis. Vamos supor que o mapa foi obtido a partir da tabela-verdade do problema (passo 1). Os quadrados estão numerados por conveniência para identificação de cada grupo.

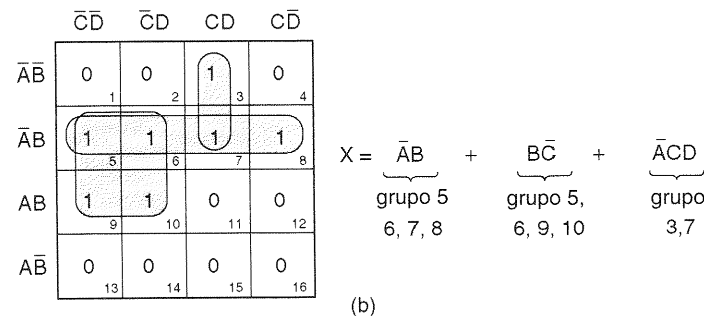
Passo 2 O quadrado 4 é o único quadrado que contém um 1 que não é adjacente a qualquer outro 1. Ele é separado e mencionado como grupo 4.

Passo 3 O quadrado 15 é adjacente *apenas* ao quadrado 11. Este par é agrupado e mencionado como grupo 11, 15.

Passo 4 Não existem octetos.

Passo 5 Os quadrados 6, 7, 10 e 11 formam um quarteto. Este quarteto é agrupado (grupo 6, 7, 10,





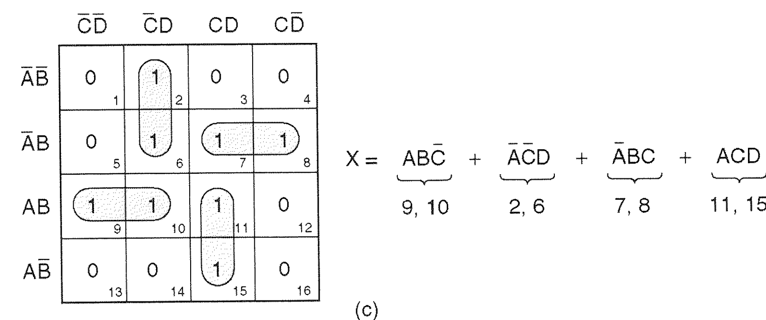


Fig. 4-15 Exemplos 4-10 até 4-12.

11). Repare que o quadrado 11 é usado novamente, embora já seja parte do grupo 11, 15.

Passo 6 Todos os 1s já estão agrupados.

Passo 7 Cada grupo gera um termo na expressão para *X*. O grupo 4 é simplesmente $\overline{A}B\overline{C}\overline{D}$. O grupo 11, 15 é *ACD* (a variável *B* foi eliminada). O grupo 6, 7, 10, 11 é *BD* (*A* e *C* foram eliminadas).

EXEMPLO 4-11

Considere o mapa K na Fig. 4-15(b). Mais uma vez presumimos que o passo 1 já foi realizado.

Passo 2 Não existem 1s isolados.

Passo 3 O 1 no quadrado 3 é adjacente *apenas* ao 1 do quadrado 7. Agrupando-se este par (grupo 3, 7), produz-se o termo $\overline{A}CD$.

Passo 4 Não existem octetos.

Passo 5 Existem dois quartetos. Os quadrados 5, 6, 7 e 8 formam um quarteto. Reunindo-se este quarteto produz-se o termo $\overline{A}B$. O segundo quarteto é formado pelos quadrados 5, 6, 9 e 10. Este quarteto é agrupado porque contém dois quadrados que não tinham sido combinados anteriormente. Este grupo produz $B\overline{C}$.

Passo 6 Todos os 1s já estão agrupados.

Passo 7 Os termos gerados pelos três grupos são unidos por um OR para obtermos a expressão para *X*.

EXEMPLO 4-12

Considere o mapa K na Fig. 4-15(c).

Passo 2 Não existem 1s isolados.

Passo 3 O 1 no quadrado 2 é adjacente apenas ao 1 no quadrado 6. Este par é agrupado para produzir $\overline{A}\overline{C}D$. Analogamente, o quadrado 9 é adjacente apenas ao quadrado 10. Combinando-se este par produz-se $AB\overline{C}$. Do mesmo modo, o grupo 7, 8 e o grupo 11, 15 produzem os termos $\overline{A}BC$ e *ACD*, respectivamente.

Passo 4 Não existem octetos.

Passo 5 Existe um quarteto formado pelos quadrados 6, 7, 10 e 11. Este quadrado, no entanto, *não* é combinado, porque todos os 1s no quarteto já foram incluídos em outros grupos.

Passo 6 Todos os 1s já foram agrupados.

Passo 7 A expressão para *X* está mostrada na figura.

EXEMPLO 4-13

Considere o mapa K na Fig. 4-16(a).

Passo 2 Não existem 1s isolados.

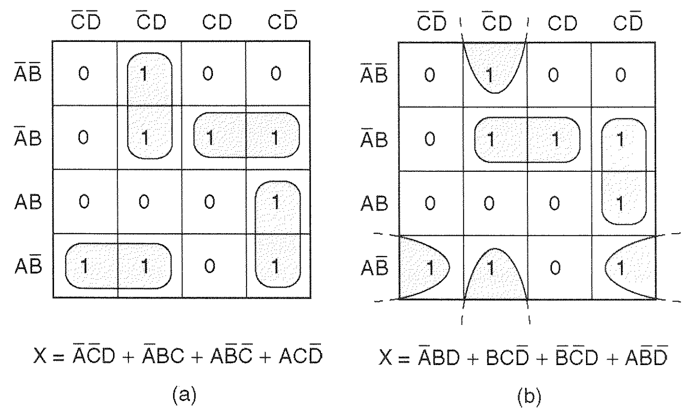


Fig. 4-16 O mesmo mapa K com duas soluções igualmente boas.

Passo 3 Não existe nenhum 1 que seja adjacente a apenas um outro 1.

Passo 4 Não existem octetos.

Passo 5 Não existem quartetos.

Passos 6 e 7 Existem muitos pares possíveis. O processo de agrupar deve usar o mínimo número de grupos para envolver todos os 1s. Para este mapa existem *duas* possibilidades, que requerem apenas quatro pares envolvidos. A Fig. 4-16(a) mostra uma solução e sua expressão resultante. A Fig. 4-16(b) mostra a outra. Note que ambas as expressões têm a mesma complexidade, e portanto nenhuma é melhor do que a outra.

EXEMPLO 4-14

Utilize o mapa K para simplificar a expressão $y = \overline{A}\overline{B}\overline{C} + \overline{B}C + \overline{A}B$.

Solução

Neste problema não é apresentada uma tabela-verdade para o preenchimento do mapa K. Em vez disso, devemos preencher o mapa K tomando cada um dos termos produto na expressão e colocando 1s nos quadrados correspondentes.

O primeiro termo, $\overline{A}\overline{B}\overline{C}$, indica que um 1 deve ser colocado no quadrado $\overline{A}\overline{B}\overline{C}$ do mapa (veja a Fig. 4-17). O segundo termo, $\overline{B}C$, indica que um 1 deve ser colocado em cada quadra-

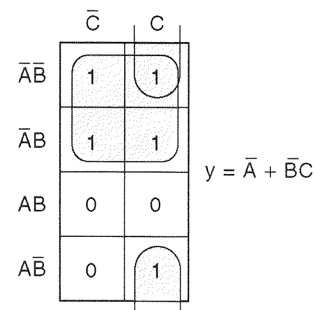


Fig. 4-17 Exemplo 4-14.

do que contém $\overline{B}C$ no seu rótulo. Na Fig. 4-17, isto acontece nos quadrados $A\overline{B}C$ e $\overline{A}\overline{B}C$. Do mesmo modo, o termo $\overline{A}B$ indica que 1 deve ser colocado nos quadrados $\overline{A}BC$ e $\overline{A}B\overline{C}$. Todos os outros quadrados devem ser preenchidos com 0s.

Agora o mapa K pode ser usado para simplificação. O resultado é $y = \overline{A} + \overline{B}C$, como apresentado na figura.

Condições "Don't Care"

Alguns circuitos lógicos podem ser projetados, de modo que existam certas condições de entrada para as quais não existam níveis de saída especificados, usualmente porque estas condições de entrada nunca ocorrerão. Em outras palavras, existem certas combinações de níveis de entrada em que "não importa" (do inglês "*don't care*") se a saída está em ALTO ou BAIXO. Isto está ilustrado na tabela-verdade da Fig. 4-18(a).

Aqui a saída z não está especificada nem como 0 nem como 1 para as seguintes condições: A, B, C = 1, 0, 0 e A, B, C = 0, 1, 1. Em vez disso, um x é mostrado para estas condições. O x representa a ***condição don't care***. Uma condição *don't care* pode surgir por várias razões; a mais comum é a existência de algumas situações nas quais certas combinações de entrada não podem nunca ocorrer, e portanto não existe saída especificada para estas condições.

Um projetista de circuitos está livre para fazer a saída ser 0 ou 1 para qualquer condição *don't care*, de modo a produzir a expressão de saída mais simples. Por exemplo, o mapa K para esta tabela-verdade é mostrado na Fig. 4-18(b) com um x colocado nos quadrados $A\overline{B}\overline{C}$ e $\overline{A}BC$. Neste caso, o projetista deve ser inteligente para substituir o x no quadrado $A\overline{B}\overline{C}$ por 1 e o x no quadrado $\overline{A}BC$ por 0, já que isto produz um quarteto que pode ser agrupado para resultar em $z = A$, conforme mostra a Fig. 4-18(c).

Toda vez que condições *don't care* ocorrem, devemos decidir qual x deve mudar para 0 e qual deve mudar para 1, de modo a produzir o melhor grupo no mapa K (isto é, a expressão mais simples). Esta decisão nem sempre é fácil. Muitos problemas no fim do capítulo proporcionarão prática em lidar com casos de *don't care*.

Resumo

O processo do mapa K tem muitas vantagens sobre o método algébrico. O mapa K é um processo mais ordenado, com passos bem-definidos quando comparado com o processo de tentativa e erro, algumas vezes usado na simplificação algébrica. Usualmente, o mapa K necessita de menos etapas, sobretudo para expressões que contêm muitos termos, e ele sempre produz uma expressão mínima.

Contudo, alguns professores preferem o método algébrico porque ele requer um amplo conhecimento da álgebra booleana e não é apenas um procedimento mecânico. Cada método tem suas vantagens, e, embora a maioria dos projetistas de lógica seja adepta dos dois, ser competente em um método é o suficiente para produzir resultados aceitáveis.

Existem outras técnicas mais complexas que os projetistas usam para minimizar circuitos lógicos. Estas técnicas são especialmente apropriadas para circuitos com um grande número de entradas, nos quais tanto o método algébrico quanto o do mapa K são impraticáveis. A maioria dessas técnicas pode ser transformada em um programa de computador que realiza a minimização sobre a tabela-verdade ou sobre a expressão completa.

Questões de Revisão

1. Use o mapa K para simplificar a expressão do Exemplo 4-7.
2. Use o mapa K para simplificar a expressão do Exemplo 4-8. Isto deve enfatizar a vantagem de utilização do mapa K para expressões que contêm muitos termos.
3. Simplifique a expressão do Exemplo 4-9 usando um mapa K.
4. O que é uma condição *don't care*?

4-6 CIRCUITOS *EXCLUSIVE*-OR E *EXCLUSIVE*-NOR

Dois circuitos lógicos especiais que freqüentemente aparecem em sistemas digitais são os circuitos *exclusive-OR* e o *exclusive-NOR*.

Exclusive-OR

Considere o circuito lógico da Fig. 4-19(a). A expressão de saída deste circuito é

$$x = \overline{A}B + A\overline{B}$$

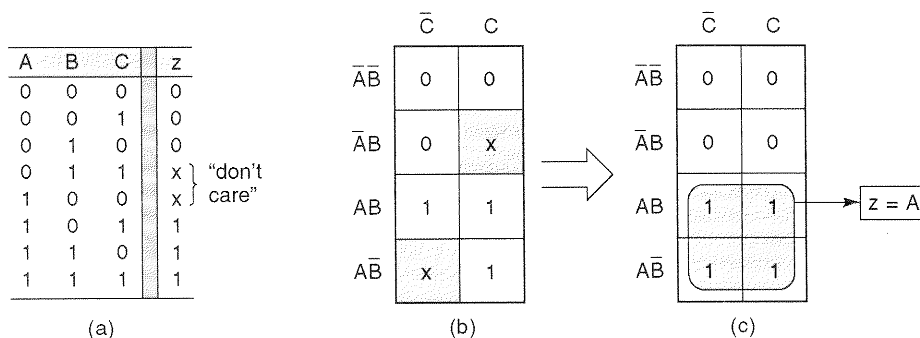
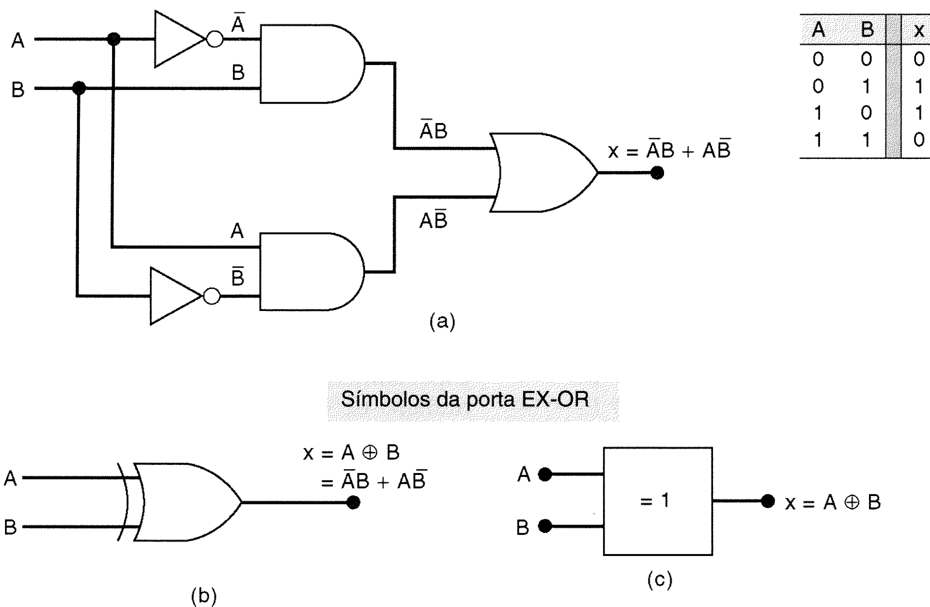


Fig. 4-18 Condições *don't care* podem ser substituídas por 0 ou 1 para produzir o grupo que resulta na expressão mais simples.



A	B	x
0	0	0
0	1	1
1	0	1
1	1	0

Fig. 4-19 (a) Tabela-verdade e circuito *exclusive*-OR; (b) símbolo tradicional da porta EX-OR; (c) símbolo IEEE/ANSI para a porta EX-OR.

A tabela-verdade apresentada mostra que $x = 1$ para dois casos: $A = 0$, $B = 1$ (o termo $\overline{A}B$) e $A = 1$, $B = 0$ (o termo $A\overline{B}$). Em outras palavras:

> **Este circuito produz uma saída em ALTO sempre que as duas entradas estão em níveis opostos.**

Este é o circuito ***exclusive*-OR**, que daqui para a frente será abreviado como **EX-OR**.

Essa combinação especial de portas lógicas ocorre freqüentemente e é muito útil em certas aplicações. Na verdade, o circuito EX-OR tem um símbolo próprio, que é mostrado na Fig. 4-19(b). Supõe-se que este símbolo contém todas as portas lógicas de um circuito EX-OR e portanto tem a mesma expressão lógica e a mesma tabela-verdade. Esse circuito EX-OR é normalmente mencionado como uma *porta* EX-OR, que é considerada um outro tipo de porta lógica. O símbolo IEEE/ANSI para uma porta EX-OR é mostrado na Fig. 4-19(c). A notação de dependência (= 1) dentro do bloco indica que a saída está ativa-ALTO *somente* quando uma única entrada está em ALTO.

Uma porta EX-OR tem apenas *duas entradas*. Não existem portas EX-OR de três ou quatro entradas. As duas entradas são combinadas de modo que $x = \overline{A}B + A\overline{B}$. Um modo abreviado que algumas vezes é usado para indicar uma expressão de saída EX-OR é

$$x = A \oplus B$$

onde o símbolo $\oplus$ representa a operação da porta EX-OR.

As características de uma porta EX-OR podem ser resumidas como se segue:

1. Tem apenas duas entradas e sua saída é

$$x = \overline{A}B + A\overline{B} = A \oplus B$$

2. Sua saída está em ALTO somente quando as duas entradas estão em níveis *diferentes*.

Diversos CIs que contêm portas EX-OR estão disponíveis. Os chips relacionados a seguir são EX-OR *quádruplos*, que contêm quatro portas EX-OR.

- **74LS86** EX-OR quádruplo (família TTL)
- **74C86** EX-OR quádruplo (família CMOS)
- **74HC86** EX-OR quádruplo (família HCMOS — *High-speed* CMOS — CMOS de alta velocidade)

Exclusive-NOR

O circuito ***exclusive*-NOR** (abreviado como **EX-NOR**) opera ao contrário do circuito EX-OR. A Fig. 4-20(a) mostra um circuito EX-NOR e sua respectiva tabela-verdade. A expressão de saída é

$$x = AB + \overline{A}\,\overline{B}$$

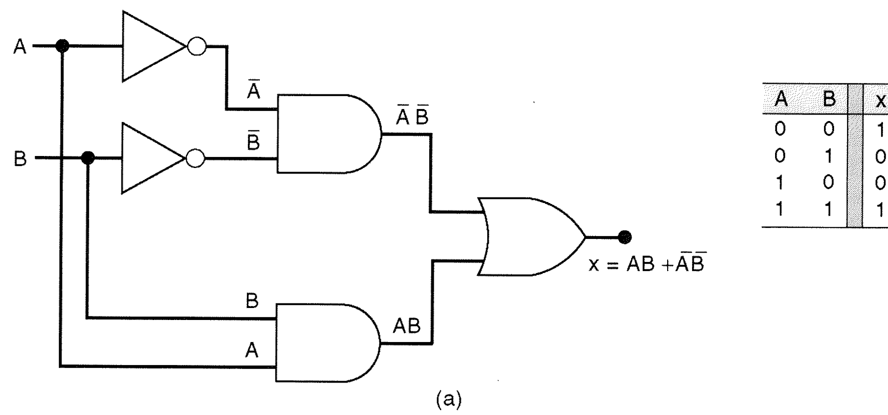
que indica juntamente com a tabela-verdade que x é 1 para dois casos: $A = B = 1$ (o termo AB) e $A = B = 0$ (o termo $\overline{A}\,\overline{B}$). Em outras palavras:

> **Este circuito produz uma saída em ALTO sempre que as duas entradas estão no mesmo nível.**

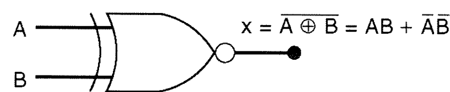
Deve estar claro que a saída de um circuito EX-NOR é exatamente o inverso da saída de um circuito EX-OR. O símbolo tradicional de uma porta EX-NOR é obtido simplesmente adicionando-se um pequeno círculo à saída do símbolo do EX-OR [Fig. 4-20(b)]. O símbolo IEEE/ANSI adiciona um pequeno triângulo à saída do símbolo EX-OR. Ambos os símbolos indicam uma saída que vai para o estado ativo em BAIXO quando *somente* uma entrada está em ALTO.

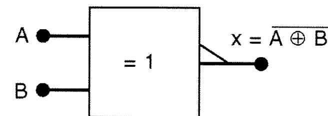
A porta EX-NOR também tem apenas *duas* entradas, e as combina de modo que sua saída é

$$x = AB + \overline{A}\,\overline{B}$$



Símbolos da porta EX-NOR





(b) (c)

Fig. 4-20 (a) Circuito *exclusive*-NOR; (b) símbolo tradicional da porta EX-NOR; (c) símbolo IEEE/ANSI.

Um modo abreviado de indicar uma expressão de saída de um EX-NOR é

$$x = \overline{A \oplus B}$$

que é simplesmente o inverso da operação EX-OR. A porta EX-NOR é resumida como se segue:

1. Tem apenas duas entradas e sua saída é

$$x = AB + \bar{A}\bar{B} = \overline{A \oplus B}$$

2. Sua saída está em ALTO somente quando as duas entradas estão no *mesmo* nível.

Diversos CIs que contêm portas EX-NOR estão disponíveis. Os chips relacionados a seguir são EX-NOR quádruplos, que contêm quatro portas EX-NOR.

- **74LS266** EX-NOR quádruplo (família TTL)
- **74C266** EX-NOR quádruplo (família CMOS)
- **74HC266** EX-NOR quádruplo (família HCMOS)

Cada um desses chips EX-NOR, entretanto, tem um circuito especial de saída que limita seu uso a certos tipos de aplicações. Muito freqüentemente, um projetista obtém a função EX-NOR simplesmente conectando a saída de um EX-OR a um INVERSOR.

EXEMPLO 4-15

Determine a forma de onda da saída para as formas de onda de entrada na Fig. 4-21.

Solução

A forma de onda da saída é obtida sabendo que a saída EX-OR vai para ALTO somente quando suas entradas têm níveis diferentes. A forma de onda resultante revela vários pontos interessantes:

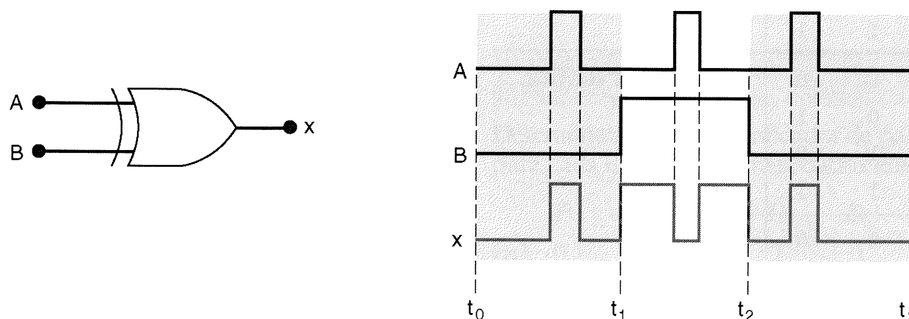


Fig. 4-21 Exemplo 4-15.

1. A forma de onda de x segue a forma de onda da entrada A durante os intervalos de tempo em que $B = 0$. Isto ocorre durante os intervalos de tempo t_0 a t_1 e t_2 a t_3.
2. A forma de onda de x é o *inverso* da forma de onda da entrada A durante os intervalos de tempo em que $B = 1$. Isto ocorre durante o intervalo t_1 a t_2.
3. Essas observações mostram que uma porta EX-OR pode ser usada como um *INVERSOR controlado*, isto é, uma de suas entradas pode ser utilizada para controlar se o sinal presente na outra entrada deve ou não ser invertido. Esta propriedade é muito útil em certas aplicações.

EXEMPLO 4-16

x_1x_0 representa um número binário de dois bits que pode ter qualquer valor (00, 01, 10 ou 11); por exemplo, quando $x_1 = 1$ e $x_0 = 0$, o número binário é 10, e assim por diante. Analogamente, y_1y_0 representa um outro número binário de dois bits. Projete um circuito lógico, usando as entradas x_1, x_0, y_1 e y_0, cuja saída vai para ALTO somente quando os dois números binários x_1x_0 e y_1y_0 são *iguais*.

Solução

O primeiro passo é construir a tabela-verdade para as 16 condições de entrada (Tabela 4-4). A saída z deve estar em ALTO sempre que os valores de x_1x_0 e y_1y_0 coincidirem, isto é, sempre que $x_1 = y_1$ e $x_0 = y_0$. A tabela mostra que existem quatro casos. Poderíamos continuar com o procedimento formal e obtermos a expressão de soma-de-produtos de z, tentarmos simplificá-la e então implementarmos o resultado. Entretanto, a natureza desse problema torna-o ideal para implementação com portas EX-NOR, e um pouco de reflexão produz uma solução simples com um mínimo esforço.

TABELA 4-4

x_1	x_0	y_1	y_0	z *(Saída)*
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

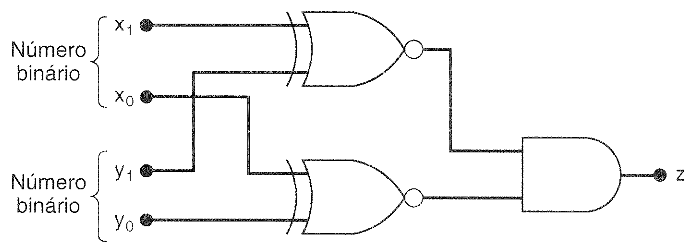


Fig. 4-22 Circuito para detectar a igualdade de dois números binários de dois bits.

No diagrama lógico da Fig. 4-22, x_1 e y_1 estão ligados a uma das portas EX-NOR, e x_0 e y_0 estão ligados à outra porta EX-NOR. A saída de cada EX-NOR está em ALTO somente quando as suas entradas são iguais. Assim, para $x_0 = y_0$ e $x_1 = y_1$, ambas as saídas das portas EX-NOR estão em ALTO. Esta é a condição procurada porque significa que os dois números de dois bits são iguais. A saída da porta AND está em ALTO somente neste caso, realizando assim a saída desejada.

EXEMPLO 4-17

Quando se simplifica a expressão para a saída de um circuito lógico combinacional, pode-se encontrar operações EX-OR ou EX-NOR durante a fatoração. Isto freqüentemente conduz ao uso de portas EX-OR ou EX-NOR na implementação do circuito final. Para ilustrar, simplifique o circuito da Fig. 4-23(a).

Solução

A expressão não-simplificada para o circuito é

$$z = ABCD + A\overline{B}\,\overline{C}D + \overline{A}\,\overline{D}$$

Podemos fatorar AD dos dois primeiros termos:

$$z = AD(BC + \overline{B}\,\overline{C}) + \overline{A}\,\overline{D}$$

À primeira vista, pode-se pensar que a expressão entre parênteses pode ser substituída por 1. Isto somente seria possível se tivéssemos $BC + \overline{BC}$. Você deveria reconhecer a expressão entre parênteses como uma combinação EX-NOR de B e C. Este fato pode ser usado para implementar novamente o circuito, conforme mostrado na Fig. 4-23(b). Este circuito é bem mais simples do que o original, pois ele usa portas com menos entradas, e dois INVERSORes foram eliminados.

Questões de Revisão

1. Utilize a álgebra booleana para provar que a expressão de saída para EX-NOR é exatamente o inverso da expressão de saída para EX-OR.
2. Qual é a saída de uma porta EX-NOR quando um sinal lógico e seu inverso estão conectados às suas entradas?
3. Um projetista necessita de um INVERSOR e tudo de que ele dispõe é uma porta EX-OR de um chip 74HC86. Ele precisa de outro chip?

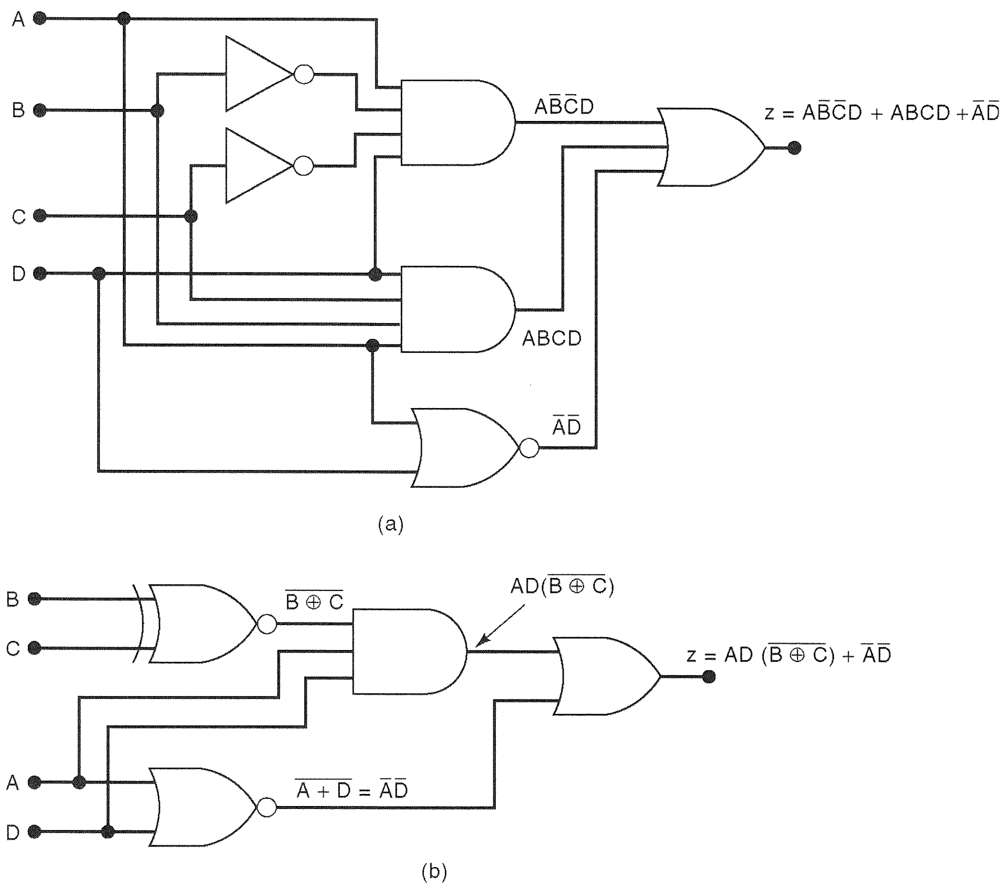


Fig. 4-23 O Exemplo 4-17 mostra como uma porta EX-NOR pode ser usada para simplificar a implementação de circuitos.

4-7 CIRCUITOS GERADOR E VERIFICADOR DE PARIDADE

No Cap. 2 vimos que um transmissor pode anexar um bit de paridade a um conjunto de bits antes de transmiti-lo para o receptor. Também foi visto como o receptor detecta qualquer erro simples em apenas um bit que possa ter ocorrido durante a transmissão. A Fig. 4-24 mostra um exemplo de circuito lógico que é usado para **geração de paridade** e **verificação de paridade**. Este exemplo em particular usa um grupo de quatro bits como sendo os dados a serem transmitidos e utiliza um bit de paridade par. Ele pode ser facilmente adaptado para utilização de paridade ímpar e para qualquer número de bits.

Na Fig. 4-24(a), os dados a serem transmitidos são aplicados ao circuito gerador de paridade que produz o bit de paridade par, *P*, como sua saída. Este bit de paridade é transmitido para o receptor junto com os bits do dado original, formando um total de cinco bits. Na Fig. 4-24(b), estes cinco bits (dado + paridade) chegam no circuito verificador de paridade do receptor, que produz uma saída de erro, *E*, que indica se ocorreu um erro simples em um bit.

Não deve causar surpresa a utilização de portas EX-OR nesses circuitos, quando consideramos que uma porta EX-OR opera de modo a produzir uma saída em 1 se um número ímpar de suas entradas está em 1, e uma saída em 0 se um número par de suas entradas está em 1.

EXEMPLO 4-18

Determine a saída do gerador de paridade para cada um dos seguintes dados de entrada, $D_3D_2D_1D_0$: (a) 0111; (b) 1001; (c) 0000; (d) 0100. Vide Fig. 4-24(a).

Solução

Para cada caso, aplique os dados de entrada no gerador de paridade e acompanhe a saída de cada porta até a saída *P*. Os resultados são: **(a)** 1; **(b)** 0; **(c)** 0; e **(d)** 1. Note que *P* é 1 somente quando o dado original contém um número ímpar de 1s. Assim, o número total de 1s enviados ao receptor (dado + paridade) é par.

EXEMPLO 4-19

Determine a saída do verificador de paridade [vide Fig. 4-24(b)] para cada um dos seguintes dados enviados pelo transmissor:

	P	D_3	D_2	D_1	D_0
(a)	0	1	0	1	0
(b)	1	1	1	1	0
(c)	1	1	1	1	1
(d)	1	0	0	0	0

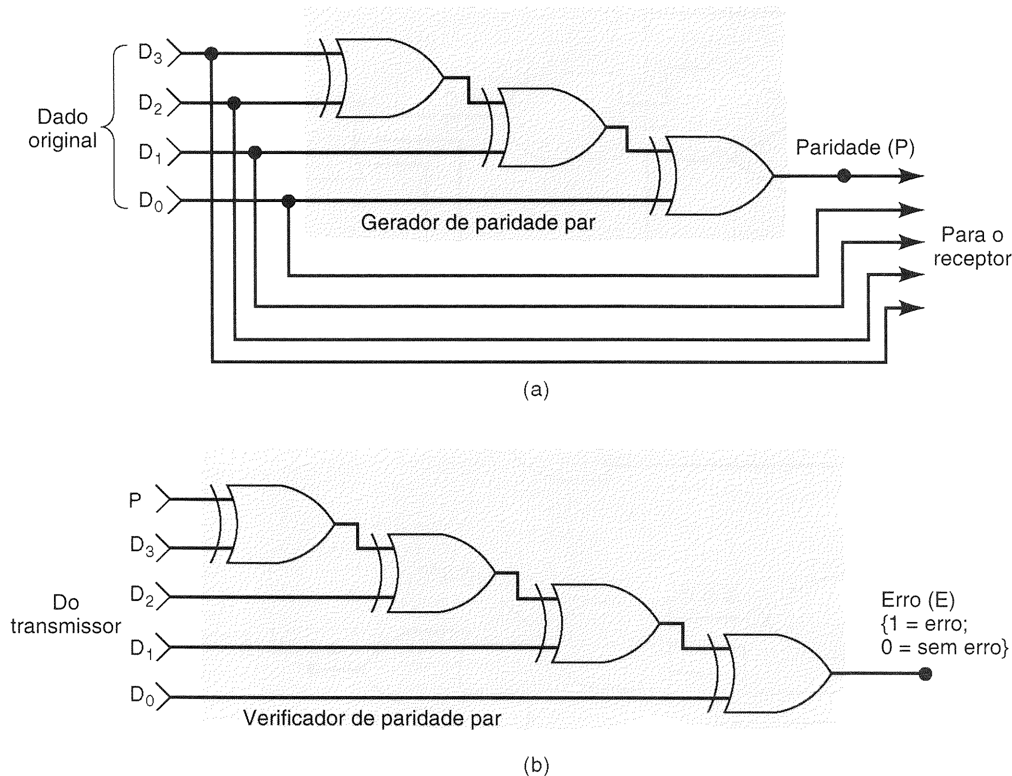


Fig. 4-24 Portas EX-OR utilizadas para implementar o gerador de paridade e o verificador de paridade para um sistema de paridade par.

Solução

Para cada caso, aplique esses níveis às entradas do verificador de paridade e siga os sinais até a saída *E*. Os resultados são: **(a)** 0; **(b)** 0; **(c)** 1; **(d)** 1. Note que 1 é produzido em *E* somente quando um número ímpar de 1s aparece nas entradas do verificador de paridade. Isto indica que ocorreu um erro, pois paridade par está sendo usada.

EXEMPLO 4-20

O circuito verificador de paridade tem como "saber" qual bit está errado?

Solução

Não. O verificador de paridade não sabe que estado cada bit de entrada deveria ter; ele sabe apenas que um número par de 1s deve estar presente. Independentemente de qual bit esteja errado, um erro simples em um bit muda o número total de 1s, de par para ímpar (ou removendo ou incluindo um bit 1), e acarreta que *E* vá para ALTO.

4-8 CIRCUITOS PARA HABILITAR/ DESABILITAR

Cada uma das portas lógicas básicas pode ser usada para controlar a passagem de um sinal lógico da entrada para a saída. Isto é detalhado na Fig. 4-25, onde um sinal lógico, *A*, é aplicado a uma entrada de cada uma das portas lógicas básicas. A outra entrada de cada porta é a entrada de controle, *B*. O nível lógico desta entrada de controle determina se o sinal de entrada está **habilitado** a alcançar a saída ou impedido (**desabilitado**) de alcançá-la. Esta ação de controle é o motivo pelo qual esses circuitos são chamados de *portas*.

Examine a Fig. 4-25 e observe que quando portas não-inversoras (AND, OR) estão habilitadas, a saída segue o sinal *A*. Ao contrário, quando portas inversoras (NAND, NOR) são habilitadas, a saída é exatamente o complemento do sinal *A*.

Repare também que portas AND e NOR produzem uma saída constante em BAIXO quando estão desabilitadas. Ao contrário, portas NAND e OR produzem uma saída constante em ALTO na condição desabilitada.

Existem muitas situações no projeto de circuitos digitais em que a passagem de um sinal lógico deve ser habilitada ou não, dependendo de condições presentes em uma ou mais entradas de controle. Muitas dessas situações são mostradas nos exemplos a seguir.

EXEMPLO 4-21

Projete um circuito lógico que permita um sinal passar para a saída somente quando as entradas de controle *B* e *C* estiverem ambas em ALTO, senão a saída deve ficar em nível BAIXO.

Solução

Uma porta AND pode ser usada porque o sinal deve ser passado sem inversão e ela produz um nível BAIXO na

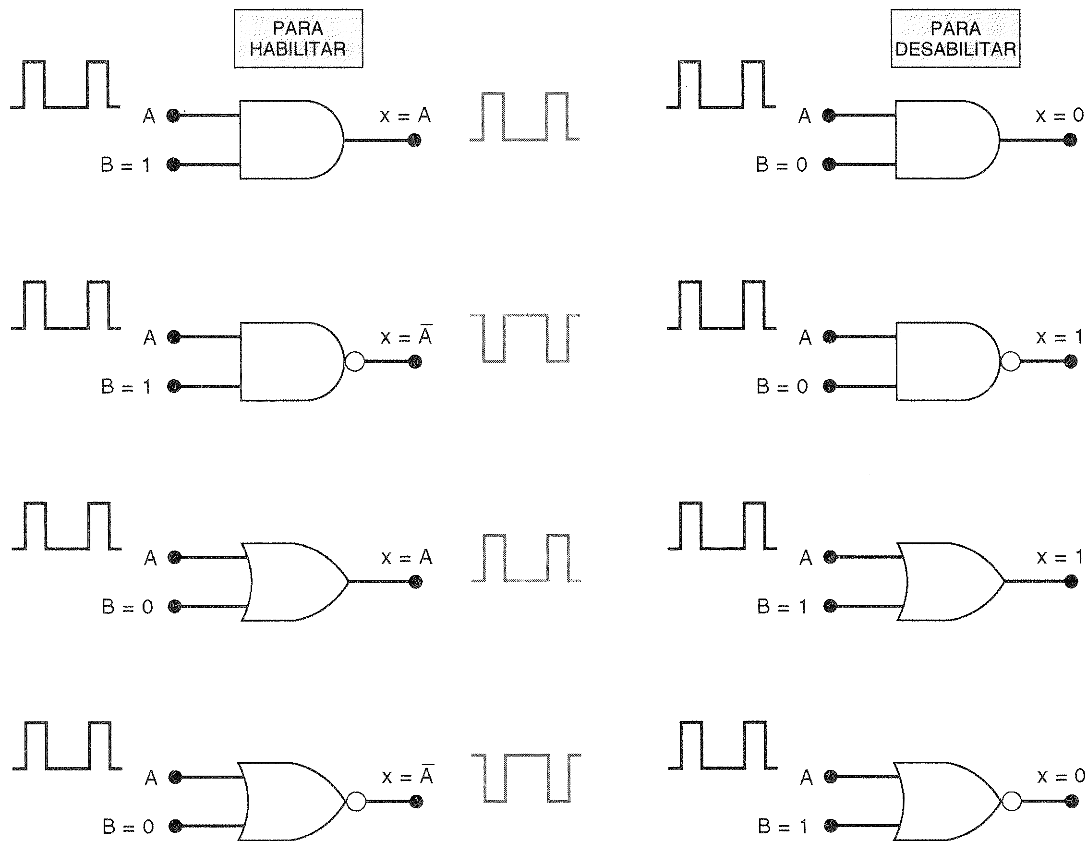


Fig. 4-25 As quatro portas básicas podem habilitar ou desabilitar a passagem de um sinal de entrada, *A*, dependendo do nível lógico da entrada de controle *B*.

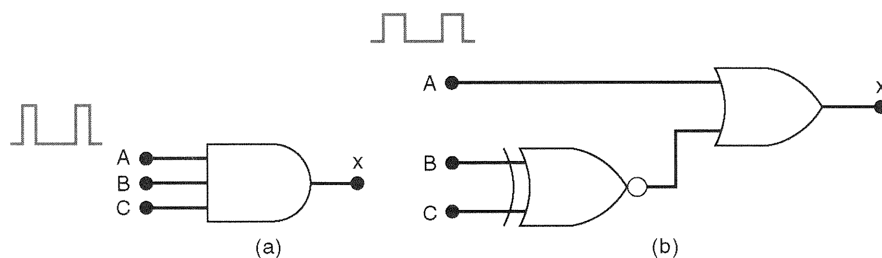


Fig. 4-26 Exemplos 4-21 e 4-22.

saída quando está desabilitada. Como a condição de habilitação deve ocorrer quando $B = C = 1$, uma porta AND de três entradas é usada, conforme ilustra a Fig. 4-26(a).

EXEMPLO 4-22

Projete um circuito lógico que permite um sinal passar para a saída somente quando uma, mas não ambas, das entradas de controle está em ALTO, caso contrário a saída fica em ALTO.

Solução

O resultado é apresentado na Fig. 4-26(b). Uma porta OR é usada porque desejamos que a saída esteja em ALTO quando a porta estiver desabilitada e não desejamos inverter o sinal. As entradas de controle *B* e *C* são combinadas numa porta EX-NOR. Quando *B* e *C* são diferentes, a porta EX-NOR envia um nível BAIXO para habilitar a porta OR. Quando *B* e *C* são iguais, o EX-NOR envia um nível ALTO para desabilitar a porta OR.

EXEMPLO 4-23

Projete um circuito lógico com um sinal de entrada *A*, uma entrada de controle *B* e saídas *X* e *Y* que opera do seguinte modo:

1. Quando $B = 1$, a saída *X* segue a entrada *A*, e a saída *Y* é 0.
2. Quando $B = 0$, a saída *X* é 0, e a saída *Y* segue a entrada *A*.

Solução

As duas saídas são 0 quando desabilitadas e seguem o sinal de entrada quando habilitadas. Assim, uma porta AND deve ser usada para cada saída. Como *X* deve ser habilitado quando $B = 1$, sua porta AND deve ser controlada por *B*, como mostra a Fig. 4-27. Tendo em vista que *Y* deve ser habilitado quando $B = 0$, sua porta AND deve ser controlada por $\overline{B}$.

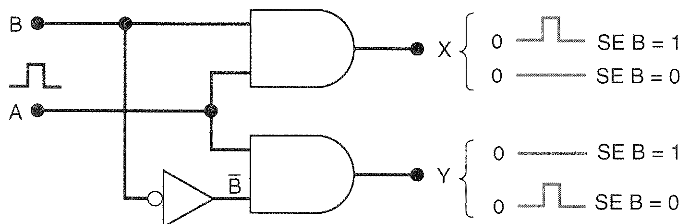


Fig. 4-27 Exemplo 4-23.

Este circuito é denominado *circuito direcionador de pulsos* porque direciona o pulso de entrada para uma ou outra saída, dependendo de *B*.

Questões de Revisão

1. Projete um circuito lógico com três entradas *A*, *B* e *C* e uma saída que vai para BAIXO somente quando *A* está ALTO enquanto *B* e *C* são diferentes.
2. Quais portas lógicas produzem uma saída em 1 quando estão desabilitadas?
3. Quais portas lógicas passam o inverso do sinal de entrada quando estão habilitadas?

4-9 CARACTERÍSTICAS BÁSICAS DE CIs DIGITAIS

CIs digitais são uma coleção de resistores, diodos e transistores fabricados em uma única peça de material semicondutor (geralmente silício), chamado de *substrato*, que é comumente conhecido pela denominação de *chip*. Este é encapsulado em uma embalagem protetora de plástico ou de cerâmica, a partir da qual saem pinos para tornar possível a ligação do CI com outros dispositivos. Um dos tipos de encapsulamento mais comum é o ***dual-in-line package* (DIP)**, mostrado na Fig. 4-28(a). Este encapsulamento tem esse nome porque contém duas linhas de pinos em paralelo. Os pinos são numerados no sentido anti-horário, a partir da marca de identificação, quando visto de cima do encapsulamento [veja Fig. 4-28(b)]. O DIP mostrado é de

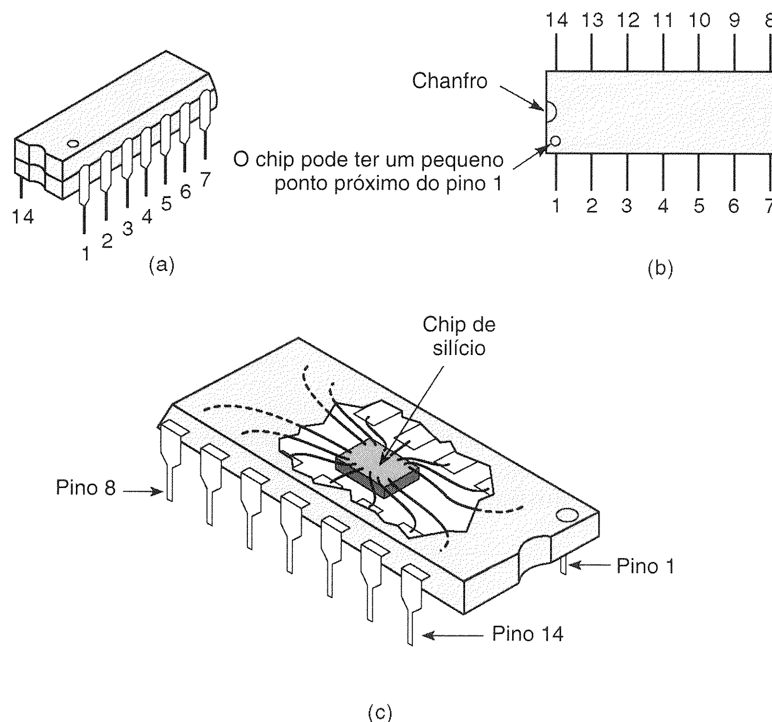


Fig. 4-28 (a) Encapsulamento *dual-in-line* (DIP); (b) vista superior; (c) o chip de silício é muito menor que o encapsulamento.

14 pinos e mede, aproximadamente, 19 mm por 6 mm. DIPs de 16, 20, 24, 28, 40 e 64 pinos também são usados.

A Fig. 4-28(c) mostra que o chip é, na verdade, muito menor que seu DIP. Ele pode ser tão pequeno quanto um quadrado de 1,2 mm. O chip de silício é conectado aos pinos do DIP através de fios bastante finos (0,025 mm de diâmetro).

O DIP é provavelmente o encapsulamento para CIs mais comum de ser encontrado em equipamentos digitais, embora outros tipos estejam se tornando cada vez mais populares. Veremos alguns desses outros tipos de encapsulamento no Cap. 8.

CIs digitais são muitas vezes classificados de acordo com a complexidade de seus circuitos, que é medida pelo número de portas lógicas equivalentes no seu substrato. Existem atualmente seis níveis de complexidade, definidos na Tabela 4-5.

Todos os CIs especificados no Cap. 3, e também neste capítulo, são chips **SSI**, que contêm um pequeno número de portas. Nos sistemas digitais modernos, dispositivos com grau médio de integração (**MSI**) e com alto grau de integração (**LSI, VLSI, ULSI** e **GSI**) realizam a maior parte das funções que antes eram implementadas por várias placas de circuito impresso cheias de chips SSI. Entretanto, eles ainda são usados como interface, ou também como lógica adicional de chips mais complexos. Geralmente, pequenas combinações de portas discretas são usadas para conectar CIs maiores entre si ou a dispositivos externos. Portanto, é necessário saber como analisar, projetar, testar e consertar circuitos combinacionais simples.

TABELA 4-5

Complexidade	Número de Portas
Small-scale integration **(SSI)**	Menor do que 12
Medium-scale integration **(MSI)**	12 a 99
Large-scale integration **(LSI)**	100 a 9.999
Very large-scale integration **(VLSI)**	10.000 a 99.999
Ultra large-scale integration **(ULSI)**	100.000 a 999.999
Giga-scale integration **(GSI)**	1.000.000 ou mais

CIs Digitais Bipolares e Unipolares

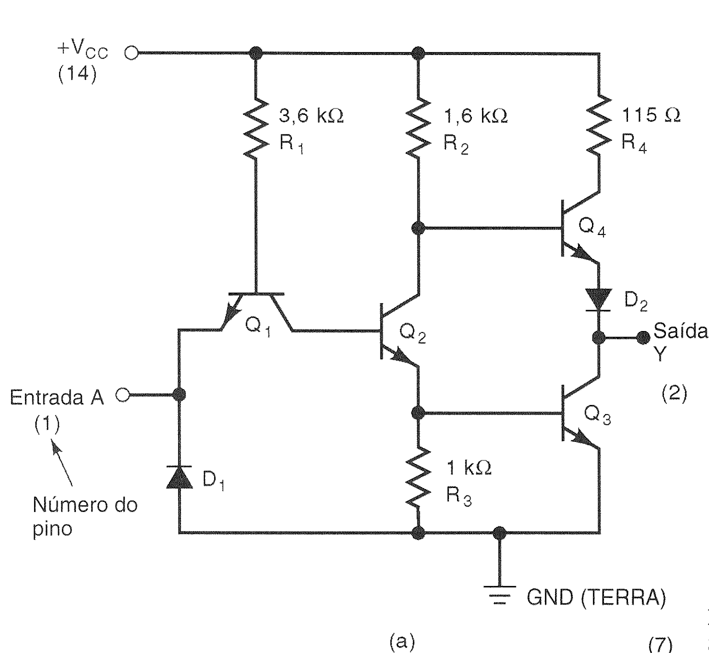
CIs digitais também podem ser classificados de acordo com o tipo de componente eletrônico usado nos seus circuitos. CIs *bipolares* são aqueles que são feitos utilizando o transistor de junção bipolar (NPN e PNP) como seu elemento principal. CIs *unipolares* são aqueles que usam transistores por efeito-de-campo (MOSFETs canal P e canal N) como seu elemento principal.

A família **TTL** (***Transistor-Transistor Logic***) é a principal família de CIs digitais bipolares nos últimos 25 anos. A Fig. 4-29(a) mostra o circuito de um INVERSOR TTL da série 74 padrão. Esta série foi a primeira dos CIs TTL. Ela não é mais utilizada em novos projetos, mas ainda é o padrão com o qual todas as outras famílias lógicas são comparadas. Observe que o circuito do INVERSOR TTL contém vários transistores bipolares, logo este tipo de transistor é o elemento principal do circuito. A família TTL é a líder nas categorias SSI e MSI há muito tempo; entretanto, essa liderança vem sendo ameaçada pela família **CMOS** (***Complementary Metal-Oxide-Semiconductor***). Esta pertence à categoria de CIs digitais unipolares porque usa MOSFETs canal P e canal N como elemento principal do circuito. A Fig. 4-29(b) mostra o circuito de um INVERSOR CMOS padrão.

Os CIs TTL e CMOS dominam o segmento de dispositivos SSI e MSI, e portanto nos concentraremos nestas duas famílias ao longo do texto. O Cap. 8 fornecerá um estudo detalhado dos circuitos e das características de CIs TTL e CMOS. Por enquanto, precisaremos abordar apenas algumas características básicas para que possamos falar sobre pesquisa de falhas em circuitos combinacionais simples.

Família TTL

A família TTL é, na verdade, constituída de várias subfamílias ou séries. A Tabela 4-6 relaciona o nome de cada uma das séries com o prefixo usado para identificar o CI como per-



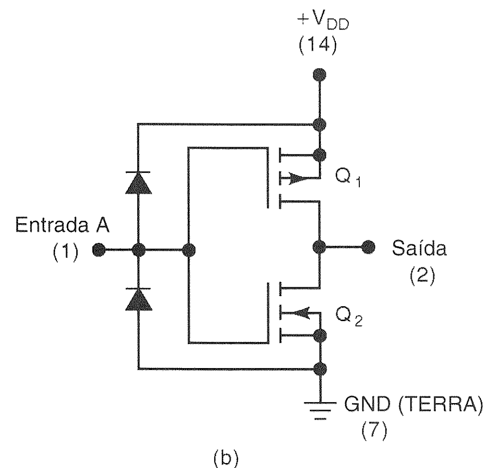


Fig. 4-29 (a) Circuito do INVERSOR TTL; (b) circuito do INVERSOR CMOS. A numeração dos pinos aparece entre parênteses.

tencente a esta série. Por exemplo, CIs que fazem parte da série TTL padrão têm um número de identificação que inicia com 74. O 7402, o 7438 e o 74123 são todos pertencentes a essa série. Do mesmo modo, CIs que pertencem à série TTL Schottky de baixa potência (*low-power Schottky*) têm o seu número de identificação começando por 74LS. O 74LS02, o 74LS38 e o 74LS123 são exemplos de dispositivos da série 74LS.

TABELA 4-6 Várias séries dentro de uma família TTL

Subfamílias TTL	Prefixo	Exemplo de CI
TTL padrão	74	7404 INVERSOR sêxtuplo
TTL Schottky	74S	74S04 INVERSOR sêxtuplo
TTL Schottky de baixa potência	74LS	74LS04 INVERSOR sêxtuplo
TTL Schottky avançada	74AS	74AS04 INVERSOR sêxtuplo
TTL Schottky avançada de baixa potência	74ALS	74ALS04 INVERSOR sêxtuplo

As diferenças entre as várias subfamílias TTL estão nas suas características elétricas, como: dissipação de potência, tempos de propagação e velocidade de comutação. Elas não diferem na disposição dos pinos ou na operação lógica realizada pelos circuitos internos. Por exemplo, o 7402, o 74S02, o 74LS02 e o 74ALS02 são todos compostos de quatro portas NOR de duas entradas. Vamos comparar as características elétricas das diferentes séries TTL no Cap. 8.

Família CMOS

Várias subfamílias CMOS disponíveis estão relacionadas na Tabela 4-7. A série 4000 é a mais antiga das séries CMOS. Ela possui muitas das funções lógicas da família TTL, mas não foi projetada para ser *compatível pino a pino* com os dispositivos TTL. Por exemplo, o chip quádruplo NOR 4001 contém quatro portas NOR de duas entradas, como o chip TTL 7402, mas as entradas e saídas das portas do chip CMOS não têm a mesma pinagem que os sinais correspondentes no chip TTL.

As séries 74C, 74HC, 74HCT, 74AC e 74ACT são as mais novas da família CMOS. As três primeiras são compatíveis pino a pino com os dispositivos TTL de mesma numeração. Por exemplo, o 74C02, 74HC02 e 74HCT02 possuem a mesma pinagem que o 7402, 74LS02, e assim por diante. As séries 74HC e 74HCT operam a uma velocidade mais alta do que os dispositivos da 74C. A série 74HCT é projetada para ser *eletricamente compatível* com dispositivos TTL; isto significa que um circuito integrado 74HCT pode ser diretamente conectado a dispositivos TTL sem que seja necessário nenhum circuito de interface. As séries 74AC e 74ACT são CIs de altíssimo desempenho. Nenhum deles é compatível pino a pino com TTL. Dispositivos 74ACT são eletricamente compatíveis com TTL. Exploraremos as várias subfamílias TTL e CMOS com bastante detalhes no Cap. 8.

Alimentação e Terra

Para utilizar CIs digitais, é necessário que se façam as conexões apropriadas aos pinos do CI. As conexões mais importantes são as de *alimentação* e *terra*. Estas conexões são necessárias para que o chip opere de modo correto. Observando a Fig. 4-29, podemos ver que tanto os circuitos TTL quanto os CMOS têm a fonte de alimentação ligada a um pino e a terra conectada em outro. V_{CC} é o nome dado ao pino no qual conectamos a alimentação em circuitos TTL. Nos circuitos CMOS, este pino é chamado de V_{DD}. Uma vez que muitos circuitos integrados CMOS são projetados para serem compatíveis com circuitos TTL, V_{CC} também é usado para designar o pino da fonte de alimentação.

Caso a ligação com a fonte de alimentação ou com terra não seja feita, as portas lógicas no chip não vão responder de modo correto às entradas lógicas, e ele não fornecerá os níveis lógicos de saída esperados.

Faixas de Tensão para os Níveis Lógicos

Para dispositivos TTL, V_{CC} deve ser de +5 V. Para dispositivos CMOS, V_{DD} pode estar situado na faixa que vai de +3 a +18 V, embora +5 V seja a tensão mais usada, principalmente quando dispositivos CMOS são usados em um mesmo circuito em conjunto com dispositivos TTL.

TABELA 4-7 Várias séries dentro de uma família CMOS

Subfamílias CMOS	Prefixo	Exemplo de CI
CMOS de porta metálica	40	4001 portas NOR quádruplas
CMOS de porta metálica; pinagem compatível com TTL	74C	74C02 portas NOR quádruplas
CMOS de porta de silício; alta velocidade; pinagem compatível com TTL	74HC	74HC02 portas NOR quádruplas
CMOS de porta de silício; alta velocidade; pinagem compatível com TTL; eletricamente compatível com TTL	74HCT	74HCT02 portas NOR quádruplas
CMOS avançada; pinagem incompatível com TTL; eletricamente incompatível com TTL	74AC	74AC02 portas NOR quádruplas
CMOS avançada; pinagem incompatível com TTL; eletricamente compatível com TTL	74ACT	74ACT02 portas NOR quádruplas

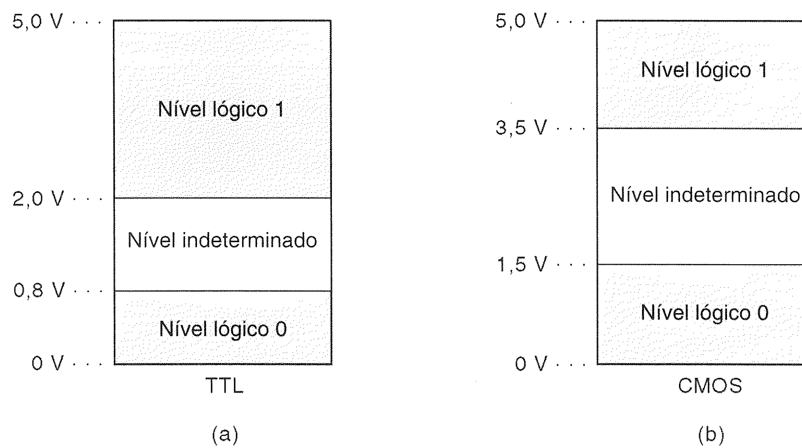


Fig. 4-30 Níveis lógicos de entrada para CIs digitais TTL e CMOS.

Para os CIs TTL padrões, as tensões de entrada aceitáveis para os níveis lógicos 0 e 1 estão definidas na Fig. 4-30(a). Um nível lógico 0 é qualquer tensão na faixa entre 0 e 0,8 V, e para o nível 1 é qualquer tensão na faixa entre 2 e 5 V. Tensões que não estão localizadas em nenhuma dessas faixas são consideradas **indeterminadas** e não devem ser usadas como entrada em nenhum dispositivo TTL. Os fabricantes de CIs não podem garantir como o circuito responderá a esses níveis que estão na faixa de indeterminação (entre 0,8 V e 2,0 V).

As faixas de tensões de entrada para circuitos integrados CMOS, que operam com V_{DD} = +5 V, podem ser vistas na Fig. 4-30(b). Tensões entre 0 e 1,5 V são definidas como nível 0, e tensões entre 3,5 e 5 V são definidas como nível 1. A faixa de indeterminação inclui tensões na faixa entre 1,5 e 3,5 V.

Entradas Não-Conectadas

O que acontece quando as entradas de um CI não estão conectadas a nenhum sinal lógico? Uma entrada desconectada é geralmente chamada de entrada em **flutuação**. A resposta para a pergunta anterior será uma para a família TTL e uma outra diferente para a CMOS.

Uma entrada TTL em flutuação funciona exatamente como se ela estivesse em nível 1. Em outras palavras, o CI vai responder como se tivéssemos aplicado um nível lógico ALTO a esta entrada. Essa característica é bastante usada quando se testa um circuito TTL. Um técnico preguiçoso poderia deixar determinadas entradas desconectadas em vez de conectá-las a um nível ALTO. Embora seja teoricamente correto, não é uma boa prática no projeto de circuitos, uma vez que uma entrada TTL em aberto é extremamente sensível a ruídos, o que pode vir a afetar o bom funcionamento do circuito.

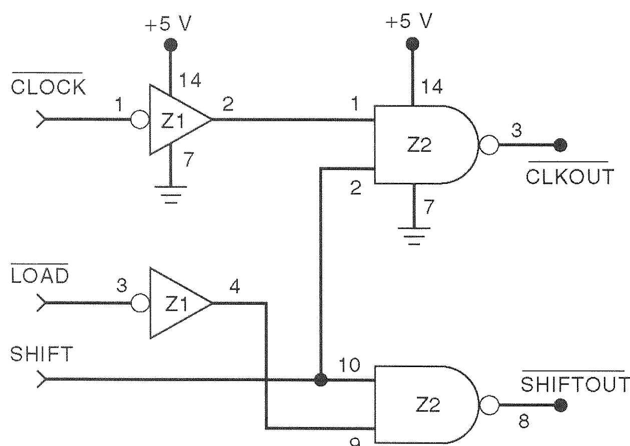
Um voltímetro ou um osciloscópio que esteja medindo uma entrada em aberto fornecerá como leitura um valor de tensão entre 1,4 e 1,8 V. Apesar de o valor lido estar situado na faixa de indeterminação para a família TTL, ele produzirá a mesma resposta que um nível 1 produziria. Lembre-se dessa característica das entradas em flutuação quando estiver pesquisando falhas em circuitos TTL.

Deixar uma entrada CMOS em flutuação pode ter resultados desastrosos. O CI pode superaquecer e possivelmente se danificar. Por essa razão, todas as entradas de circuitos integrados CMOS devem ser conectadas a um nível lógico definido (ALTO ou BAIXO) ou à saída de um outro CI. A tensão medida em uma entrada CMOS em flutuação varia em função do ruído presente, e portanto não age como um nível 0 ou 1. Isto faz com que o nível de tensão de saída oscile em função do ruído existente na entrada.

Diagramas de Circuitos Lógicos

Um diagrama de um circuito lógico mostra *todas* as conexões, a numeração dos pinos, os números dos CIs, os valores dos componentes, os nomes dos sinais e as tensões de alimentação do circuito. A Fig. 4-31 mostra um diagrama típico para um circuito lógico simples. Examine-o com cuidado e observe os seguintes pontos importantes:

1. O circuito usa portas lógicas de dois CIs diferentes. Os dois INVERSORes fazem parte do chip 74HC04, ao qual foi dada a designação Z1. O 74HC04 possui seis INVERSORes, dois dos quais são usados neste circuito, e foram associados ao chip através da designação Z1. Do mesmo modo, as portas NAND fazem parte do chip 74HC00, que contém quatro portas. Todas as portas do chip recebem a designação Z2. Referindo-se a cada porta por Z1, Z2, Z3 etc., é possível determinar a que chip pertence cada porta. Isto é essencialmente importante em circuitos mais complexos que contenham muitos CIs com muitas portas por CI.
2. O número de cada pino de entrada e de saída está indicado no diagrama. A numeração dos pinos, juntamente com a designação dos CIs, torna bastante simples a tarefa de referenciar qualquer ponto do circuito. Por exemplo, Z1, pino 2 se refere à saída do INVERSOR situado na parte superior do diagrama. Do mesmo modo, podemos dizer que Z1, pino 4 está conectado a Z2, pino 9.
3. As conexões da fonte de alimentação e do terra de cada CI são mostradas no diagrama. Por exemplo, Z1, pino 14 está conectado a +5 V e Z1, pino 7 está conectado a terra. Essas duas conexões fornecem alimentação e terra para *todos* os seis INVERSORes que fazem parte de Z1.



CI	Tipo
Z1	74HC04 INVERSOR sêxtuplo
Z2	74HC00 NAND quádruplo

Fig. 4-31 Diagrama típico de um circuito lógico.

Os fabricantes de equipamentos eletrônicos geralmente fornecem diagramas esquemáticos detalhados como o da Fig. 4-31. Esses diagramas são de grande importância quando precisamos pesquisar a causa de um problema que está ocorrendo no circuito. Escolhemos identificar os CIs como Z1, Z2, Z3 e assim por diante. Outras designações bastante usadas são CI1, CI2, CI3 etc. e U1, U2, U3 etc.

Questões de Revisão

1. Qual é o tipo mais comum de encapsulamento de CIs digitais?
2. Enumere as seis classificações existentes para a complexidade de CIs digitais.
3. *Verdadeiro* ou *falso*: Um 74S74 contém a mesma lógica e a mesma disposição de pinos que o 74LS74.
4. *Verdadeiro* ou *falso*: Um 74HC74 contém a mesma lógica e a mesma disposição de pinos que o 74AS74.
5. Quais subfamílias CMOS não são compatíveis pino a pino com a família TTL?
6. Qual é a faixa aceitável de tensão de entrada para um nível lógico 0 na família TTL? Qual é a faixa para o nível 1?
7. Repita a Questão 6 para um circuito CMOS operando com $V_{DD} = 5$ V.
8. Como um circuito TTL responde a uma entrada em flutuação?
9. Como um circuito CMOS responde a uma entrada em flutuação?
10. Quais são as subfamílias CMOS que podem ser conectadas a TTL sem que seja necessária a utilização de circuitos de interface?

4-10 PESQUISA DE FALHAS EM SISTEMAS DIGITAIS

Existem três passos básicos a serem seguidos quando estamos depurando, isto é, pesquisando problemas ou falhas em um circuito:

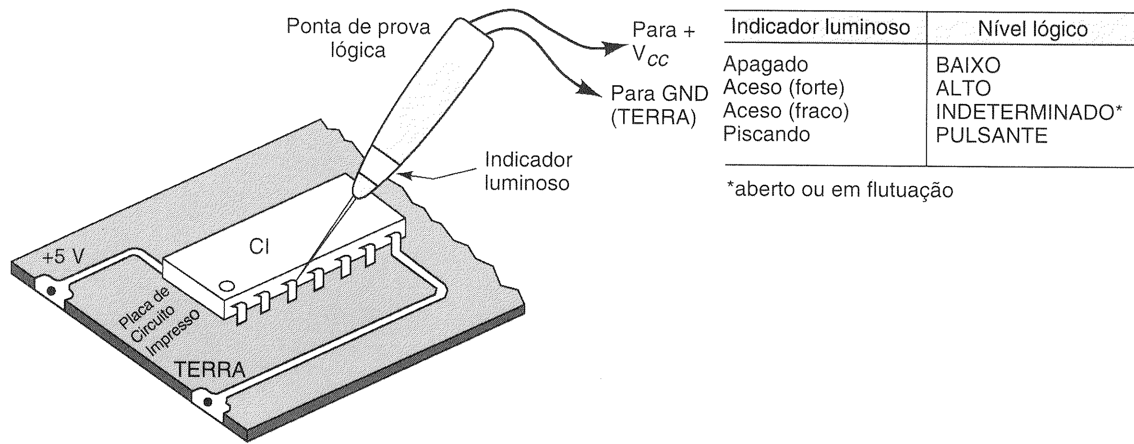
1. **Detecção da falha.** Observe a operação do circuito (ou sistema) e compare-a com a operação correta esperada.
2. **Isolamento da falha.** Realize testes e faça medições que o ajudem a isolar a falha.
3. **Correção da falha.** Troque o componente defeituoso, conserte a ligação defeituosa, remova o curto-circuito e assim por diante.

Apesar de esses passos parecerem relativamente simples, o procedimento real a ser seguido depende muito do tipo e da complexidade do circuito, das ferramentas e da documentação disponíveis.

Boas técnicas para depuração de circuitos só podem ser aprendidas em um ambiente de laboratório, através da experimentação e da prática na depuração de circuitos e sistemas reais. Não existe melhor maneira de se tornar hábil em depuração de circuitos do que praticar o máximo possível. Não importa quantos bons livros você leia, nenhum deles será capaz de lhe transmitir esse tipo de experiência. Eles podem, entretanto, ajudá-lo a desenvolver sua capacidade de análise, que é fundamental para uma depuração eficiente. Inicialmente, vamos descrever os tipos de falhas mais comuns em sistemas, que são compostos basicamente de CIs digitais, e diremos a você como reconhecê-los. Estudaremos casos típicos para ilustrar os processos de análise envolvidos na manutenção de circuitos (ou sistemas). Além disso, no final deste capítulo, apresentaremos problemas que lhe fornecerão a oportunidade de utilizar estes processos de análise, para chegar a conclusões sobre circuitos digitais defeituosos.

Para as discussões e exercícios sobre depuração que faremos neste livro, presumimos que o estudante tem acesso aos instrumentos básicos para depuração de circuitos, como por exemplo: *ponta de prova lógica, osciloscópio, gerador de pulsos, rastreador de corrente* etc. Queremos deixar bem claro que a ferramenta mais importante e eficiente na depuração é o seu cérebro, e esta é a ferramenta que esperamos desenvolver ao apresentar princípios básicos, técnicas de depuração, exemplos e problemas neste capítulo e nos seguintes.

Nas próximas três seções sobre depuração, utilizaremos apenas o seu cérebro e uma **ponta de prova lógica** como a que aparece na Fig. 4-32. As outras ferramentas serão usadas nos capítulos seguintes. A ponta de prova possui uma ponta de metal que deve tocar o ponto específico do circuito que queremos testar. A figura mostra o teste do pino



Indicador luminoso	Nível lógico
Apagado	BAIXO
Aceso (forte)	ALTO
Aceso (fraco)	INDETERMINADO*
Piscando	PULSANTE

*aberto ou em flutuação

Fig. 4-32 Uma ponta de prova lógica é utilizada para monitorar os níveis lógicos no pino do CI ou em qualquer outro ponto acessível do circuito.

3 do CI. Também podemos utilizar a ponta de prova para testar uma trilha em uma placa de circuito impresso, um fio desencapado, um pino de um conector, um terminal de um componente discreto como um transistor, ou qualquer outro ponto que seja condutor no circuito. O nível lógico presente na ponta de prova será indicado pelo estado do indicador luminoso ou LED existente na ponta de prova. Os quatro estados possíveis são mostrados na tabela da Fig. 4-32. Observe que o nível lógico *indeterminado* produz uma intensidade fraca no indicador luminoso. Isto também acontece quando a ponta de prova testa um ponto do circuito que está em aberto ou flutuando, isto é, não está conectado a nenhuma fonte de tensão.

4-11 FALHAS INTERNAS DOS CIs DIGITAIS

As falhas internas mais comuns dos CIs digitais são:

1. Mau funcionamento de circuitos internos do CI.
2. Entradas ou saídas em curto com a terra ou V_{CC}.
3. Circuito aberto nas entradas ou saídas.
4. Curto-circuito em dois pinos do CI (desde que não seja terra ou V_{CC}).

Vamos descrever cada um desses tipos de falha.

Mau Funcionamento de Circuitos Internos do CI

Este problema é causado, na maioria das vezes, quando um dos componentes internos do circuito está danificado ou opera fora de suas especificações. Quando isto acontece, as saídas do CI não respondem de modo correto às suas entradas. Não existe nenhum modo de prever qual será o comportamento da saída, uma vez que este depende do componente interno que está apresentando problema. Exemplos desse tipo de falha seriam: um curto entre a base e o emissor do transistor Q_4 ou um valor extremamente alto de resistência para R_2 no INVERSOR TTL da Fig. 4-29(a). Esse tipo de falha interna não é tão comum quanto as outras três.

Entradas Internamente em Curto com a Terra ou com a Fonte de Alimentação

Este tipo de falha interna faz com que a entrada do CI fique permanentemente em estado BAIXO ou ALTO. A Fig. 4-33(a) mostra o pino de entrada 2 da porta NAND em curto com a terra, internamente ao CI. Isto faz com que o pino 2 esteja sempre no estado BAIXO. Se um sinal lógico *B* for conectado ao pino 2, haverá um curto-circuito entre *B* e a terra. Portanto, esse tipo de falha vai afetar a saída do dispositivo que estiver gerando o sinal *B*.

Do mesmo modo, um pino de entrada de um CI poderia estar internamente em curto com +5 V, como na Fig. 4-33(b). Isto faz com que este pino esteja sempre no nível ALTO. Se esta entrada for conectada a um sinal lógico *A*, haverá um curto entre *A* e +5 V.

Saídas Internamente em Curto com a Terra ou com a Fonte de Alimentação

Este tipo de falha faz com que o pino de saída fique permanentemente no estado BAIXO ou ALTO. A Fig. 4-33(c) mostra o pino 3 da porta NAND internamente em curto com a terra. Esta saída estará sempre em BAIXO e não responderá as condições aplicadas aos pinos de entrada 1 e 2. Em outras palavras, as entradas *A* e *B* não terão efeito sobre a saída *X*.

Um pino de saída de um CI também pode estar internamente em curto com +5 V, como podemos ver na Fig. 4-33(d). Isto faz com que o pino de saída 3 esteja em ALTO independentemente do estado dos sinais nos pinos de entrada. Observe que este tipo de falha não afeta os sinais lógicos presentes nas entradas.

EXEMPLO 4-24

Observe o circuito da Fig. 4-34. Um estudante pode utilizar uma ponta de prova para determinar as condições existentes em vários pinos do CI. Os resultados estão mostrados na tabela da figura. Examine estes resultados e determine

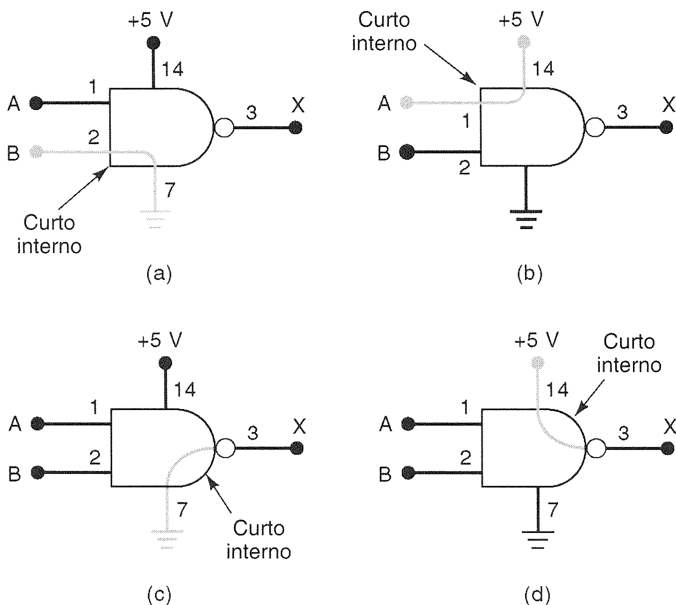


Fig. 4-33 (a) Entrada do CI internamente em curto com a terra; (b) entrada do CI internamente em curto com a fonte de alimentação. Esses dois tipos de falha forçam o sinal de entrada que está neste pino a ficar no mesmo estado. (c) Saída de um CI internamente em curto com a terra; (d) saída de um CI internamente em curto com a fonte de alimentação. Esses dois tipos de falha não afetam os sinais que estão nas entradas do CI.

se o circuito está operando corretamente. Caso não esteja, indique algumas das possíveis falhas.

Solução

O pino de saída 4 do INVERSOR deveria estar pulsando, uma vez que a sua entrada está. Entretanto, os resultados mostram que o pino 4 está sempre em BAIXO. Este pino está conectado ao pino 1 de Z2 e portanto faz com que a saída da porta NAND seja ALTA. A partir desta discussão, podemos enumerar três possíveis falhas que causariam esse comportamento.

A primeira seria devido à existência de um componente interno danificado, que impediria que o INVERSOR funcionasse corretamente. A segunda seria devido a um curto-circuito interno entre o pino 4 e o terra de Z1. Isto faria com que o pino 4 permanecesse sempre em BAIXO, e finalmente a terceira seria o resultado de um curto-circuito interno entre o pino 1 e o terra de Z2. Isto impediria que o estado do pino de saída do INVERSOR se modificasse.

Além dessas possíveis falhas, pode haver curto-circuitos externos entre a terra e qualquer ponto na ligação entre o pino 4 de Z1 e o pino 1 de Z2. Falaremos sobre como determinar a falha presente no exemplo seguinte.

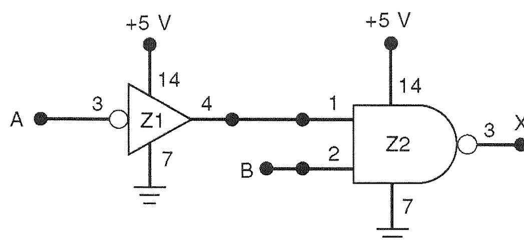
Circuito Aberto nas Entradas ou Saídas

Às vezes é possível que o fio extremamente fino que liga o pino do CI ao seu circuito interno se quebre, produzindo um circuito aberto. A Fig. 4-35 mostra esta situação para uma entrada (pino 13) e para uma saída (pino 6). Se o sinal for aplicado ao pino 13, ele não alcançará a entrada da porta NAND 1 e, portanto, não terá efeito sobre a saída da NAND 1. O circuito aberto deixa a entrada em flutuação. Como foi dito anteriormente, dispositivos TTL responderão a uma entrada em flutuação como se ela tivesse um nível lógico 1. Já os dispositivos CMOS vão responder de modo imprevisível e podem até ser danificados por superaquecimento.

Um circuito aberto na saída do NAND 4 impede que este sinal chegue ao pino 6, e portanto não há uma tensão estável presente neste pino. Se ele for conectado a uma entrada de um outro CI, ele vai produzir uma condição de flutuação nesta entrada.

EXEMPLO 4-25

O que uma ponta de prova lógica indicaria no pino 13 e no pino 6 do CI da Fig. 4-35?



Pino	Condição
Z1-3	Pulsante
Z1-4	BAIXO
Z2-1	BAIXO
Z2-2	ALTO
Z2-3	ALTO

Fig. 4-34 Exemplo 4-24.

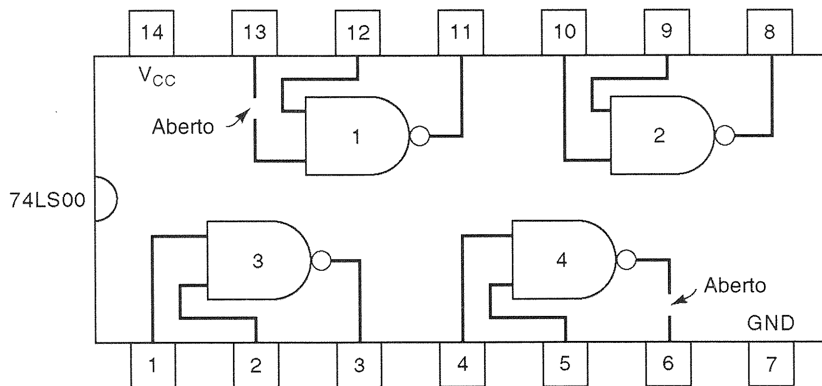


Fig. 4-35 Um CI com uma entrada desconectada internamente não responderá aos sinais aplicados naquele pino de entrada. Uma saída internamente desconectada vai produzir uma tensão imprevisível neste pino de saída.

Solução

No pino 13, a ponta de prova indicaria o nível lógico do sinal externo que está conectado ao pino 13 (e que não aparece na figura). No pino 6, a ponta de prova indicaria uma luz fraca referente a um nível lógico indeterminado. Isto ocorre porque o nível de saída da porta NAND não chega até o pino 6.

EXEMPLO 4-26

Observe o circuito da Fig. 4-36 e as anotações feitas das indicações da ponta de prova. Quais são as possíveis falhas que poderiam causar essas indicações? Suponha que os CIs são TTL.

Solução

Examinando as anotações feitas, podemos verificar que o INVERSOR está funcionando corretamente, mas a saída da porta NAND está incompatível com suas entradas. A saída da porta NAND deveria estar em ALTO, uma vez que o pino de entrada 1 está em BAIXO. Este nível BAIXO deveria impedir que a porta NAND respondesse aos pulsos no pino 2. É provável que este nível BAIXO não esteja chegando ao circuito interno da porta NAND, em virtude de a entrada estar internamente em aberto. Como este CI é TTL, esse circuito aberto iria produzir o mesmo efeito de um nível ALTO no pino 1. Se o CI fosse CMOS, este circuito aberto interno no pino 1 poderia produzir um nível indeterminado na saída e possivelmente causar a destruição do CI por superaquecimento.

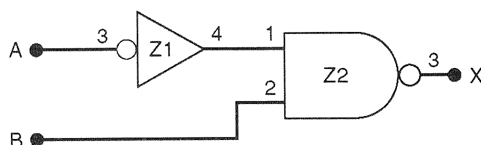
A partir de afirmações feitas anteriormente sobre entradas TTL em aberto, você poderia esperar que a tensão no pino 1 de Z2 estivesse entre 1,4 e 1,8 V, e isto deveria ter sido indicado como um nível indeterminado pela ponta de prova. Isto seria verdadeiro se o circuito aberto fosse *externo* ao chip NAND. Não existe circuito aberto entre o pino 4 de Z1 e o pino 1 de Z2, e portanto a tensão existente no pino 4 de Z1 chega até o pino 1 de Z2, logo o circuito está aberto no *interior* do chip.

Curto-Circuito entre Dois Pinos

Um curto interno entre dois pinos de um CI obriga que os sinais lógicos nestes pinos sejam sempre idênticos. Sempre que dois sinais supostamente diferentes apresentarem as mesmas variações em seus níveis lógicos, é provável que estes sinais estejam em curto.

Considere o circuito da Fig. 4-37, onde os pinos 5 e 6 de uma porta NOR estão em curto por dentro do CI. Este curto faz com que os pinos de saídas dos INVERSORes estejam conectados, obrigando que os sinais dos pinos Z1-2 e Z1-4 sejam idênticos, mesmo quando seus respectivos sinais de entrada estejam tentando produzir saídas diferentes. Para ilustrar, considere as formas de onda das entradas mostradas no diagrama. Apesar de estas formas de onda serem diferentes, as formas de onda nas saídas Z1-2 e Z1-4 são as mesmas.

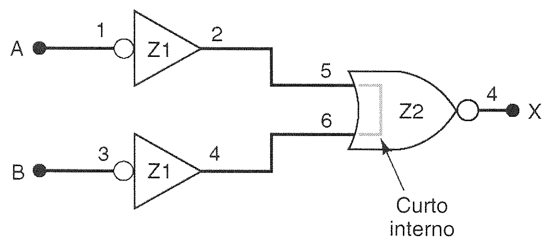
Durante o intervalo de tempo entre t_1 e t_2, os dois INVERSORes têm suas entradas em ALTO e portanto estão tentando produzir saídas em BAIXO. Logo, o fato de as saídas estarem em curto não afeta o resultado. Entretanto, durante os intervalos de t_2 até t_3 e de t_3 até t_4, um dos INVERSORes



Nota: As ligações dos CIs a V_{cc} e a terra foram omitidas.

Pino	Condição
Z1-3	ALTO
Z1-4	BAIXO
Z2-1	BAIXO
Z2-2	Pulsante
Z2-3	Pulsante

Fig. 4-36 Exemplo 4-26.



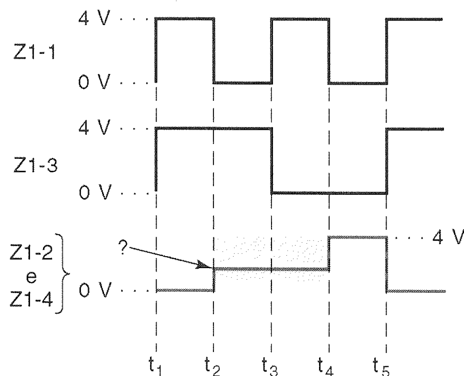


Fig. 4-37 Quando dois pinos estão internamente em curto, os sinais que estão conectados a esses pinos são forçados a ser idênticos, possuindo geralmente três níveis distintos.

está tentando fazer com que sua saída vá para ALTO, enquanto o outro está tentando fazer com que sua saída vá para BAIXO. Nessa situação, a tensão que aparece de fato nas saídas em curto depende dos circuitos internos dos CIs. Para dispositivos TTL, esta tensão estará, geralmente, próxima ao extremo superior da faixa que determina o nível 0 (isto é, próxima a 0,8 V), embora também seja possível que ela possa estar na faixa de indeterminação. Para o caso de dispositivos CMOS, a tensão estará geralmente na faixa para a qual o nível lógico é indeterminado.

Sempre que você vir uma forma de onda como a do sinal Z1-2 e Z1-4 na Fig. 4-37, isto é, como três níveis diferentes, você deve suspeitar de que os dois sinais de saída podem estar em curto.

Questões de Revisão

1. Enumere as diferentes falhas internas dos CIs digitais.
2. Que tipo de falha interna pode produzir sinais como aqueles que contêm três níveis diferentes de tensão?
3. O que a ponta de prova indicaria em Z1-2 e Z1-4 da Fig. 4-37 se $A = 0$ e $B = 1$?

4-12 FALHAS EXTERNAS

Vimos como reconhecer os efeitos dos vários tipos de falhas internas dos CIs digitais. Muitas coisas erradas podem acontecer fora do CI, fazendo com que este não funcione corretamente. Descrevemos as mais comuns nesta seção.

Linhas de Sinal Abertas

Esta categoria inclui qualquer falha que produza uma descontinuidade no caminho elétrico, de tal modo que o nível de tensão (ou sinal) seja impedido de ir de um ponto a outro. Algumas das causas de linhas de sinal abertas são:

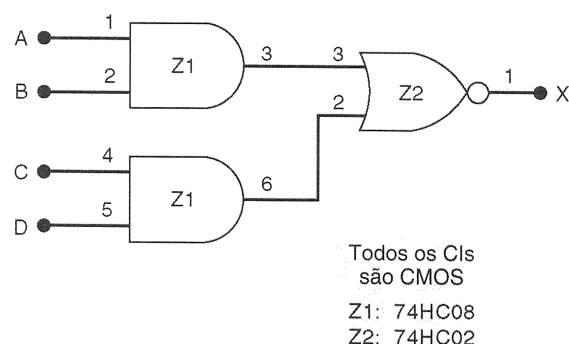
1. Fio partido.
2. Conexão com solda fria ou conexão com *wire-wrap** frouxa.
3. Cortes ou fissuras nas trilhas do circuito impresso (alguns são tão pequenos que são difíceis de ver sem o auxílio de uma lente de aumento).
4. Pino do CI dobrado ou quebrado.
5. Mau contato no soquete do CI.

Esse tipo de falha no circuito pode ser descoberto através de uma inspeção visual cuidadosa e de uma posterior verificação da continuidade (isto é, da baixa resistência em um caminho elétrico), desconectando a fonte de alimentação e colocando o ohmímetro entre os dois pontos em questão.

EXEMPLO 4-27

Considere o circuito CMOS da Fig. 4-38 e as indicações da ponta de prova. Qual é a falha mais provável deste circuito?

*Técnica de montagem de circuitos digitais utilizada principalmente na construção de protótipos. Os CIs são montados em soquetes especiais de pinos longos sobre os quais os fios são enrolados (daí o nome *wire-wrap*). (*N. T.*)



Pino	Condição
Z1-1	Pulsante
Z1-2	ALTO
Z1-3	Pulsante
Z1-4	BAIXO
Z1-5	Pulsante
Z1-6	BAIXO
Z2-3	Pulsante
Z2-2	Indeterminado
Z2-1	Indeterminado

Fig. 4-38 Exemplo 4-27.

Solução

O nível indeterminado na saída da porta NOR é provavelmente devido a um nível indeterminado na entrada no pino 2. Como Z1-6 está em BAIXO, este mesmo nível deveria estar presente em Z2-2. Não é difícil perceber que o nível BAIXO de Z1-6 não está chegando a Z2-2 e portanto deve haver um circuito aberto no caminho do sinal entre estes dois pontos. A localização desse circuito aberto pode ser determinada começando a rastrear com uma ponta de prova este nível BAIXO, a partir de Z1-6 e indo em direção a Z2-2. Quando a ponta de prova indicar um nível indeterminado, teremos a localização do circuito aberto.

Linhas de Sinal em Curto

Este tipo de falha causa o mesmo efeito que um curto interno entre pinos de um CI. Ele faz com que dois sinais sejam exatamente iguais. Uma linha de sinal pode ser colocada em curto com a terra ou com V_{CC}, em vez de uma outra linha de sinal. Neste caso, o sinal será forçado para um estado BAIXO ou ALTO. As principais causas para curtos inesperados entre dois pontos do circuito são:

1. **Ligações malfeitas.** Um exemplo disto é retirar uma grande parte do isolamento das pontas de fios muito próximos.
2. **Pontes de solda.** Elas nada mais são do que respingos de solda que colocam dois ou mais pontos em curto. Comumente ocorrem entre pontos que são muito próximos, como por exemplo pinos adjacentes de um chip.
3. **Corrosão incompleta.** O cobre existente entre duas trilhas adjacentes em uma placa de circuito impresso não foi completamente removido durante a corrosão da placa.

Mais uma vez, uma inspeção visual cuidadosa geralmente identifica esse tipo de falha. Além disso, uma verificação com um ohmímetro pode indicar que dois pontos no circuito estão em curto.

Falha na Fonte de Alimentação

Todos os sistemas digitais possuem uma ou mais fontes de alimentação para fornecer as tensões V_{CC} e V_{DD} de que os chips necessitam. Uma fonte de alimentação defeituosa ou em sobrecarga (suprindo mais corrente do que é capaz) fornecerá tensões de alimentação mal-reguladas aos CIs, fazendo com que eles não funcionem ou o façam de modo imprevisível.

Uma fonte de alimentação pode sair de regulação devido a uma falha em seus circuitos internos, ou porque os circuitos que ela está alimentando estão drenando mais corrente do que a fonte foi projetada para suprir. Isto ocorre se um chip ou um componente tem um defeito que faça com que ele drene muito mais corrente do que o normal.

É uma boa prática de depuração verificar se os níveis de tensão em cada fonte de alimentação do circuito estão dentro das faixas de operação especificadas. Além disso, também é uma boa idéia utilizar um osciloscópio para verificar se não há uma quantidade excessiva de *ripple* (ondulação) sobre o nível de tensão contínua e para observar se a tensão da fonte permanece regulada durante a operação do sistema.

Um dos sinais mais comuns de uma fonte defeituosa é que um ou mais CIs funcionam de modo incorreto ou simplesmente não funcionam. Alguns CIs são mais tolerantes a variações na fonte de alimentação e podem funcionar corretamente, enquanto outros não. Você sempre deve verificar os níveis de tensão nos pinos de alimentação e terra de cada CI que parece não funcionar corretamente.

Carregamento da Saída

Quando um circuito digital tem sua saída conectada a muitas entradas, sua capacidade de fornecer corrente pode ser excedida fazendo com que a tensão de saída caia na faixa para a qual o nível lógico é indeterminado. Este efeito é chamado *carregamento* do sinal (o que ocorre na verdade é uma sobrecarga do sinal de saída) e é geralmente causado por um projeto malfeito ou por ligações incorretas.

Questões de Revisão

1. Quais são os tipos de falhas externas mais comuns?
2. Enumere algumas das causas de circuitos abertos nos caminhos elétricos dos sinais.
3. Quais os sintomas causados por uma fonte de alimentação defeituosa?
4. De que modo o carregamento pode afetar o nível de tensão de um CI?

4-13 ESTUDO DE UM CASO DE PESQUISA DE FALHAS

O exemplo seguinte ilustra o processo de análise envolvido na depuração de circuitos digitais. Embora este exemplo seja de um circuito lógico combinacional bastante simples, o raciocínio e os procedimentos utilizados podem ser aplicados a circuitos digitais mais complexos, que encontraremos nos capítulos seguintes.

EXEMPLO 4-28

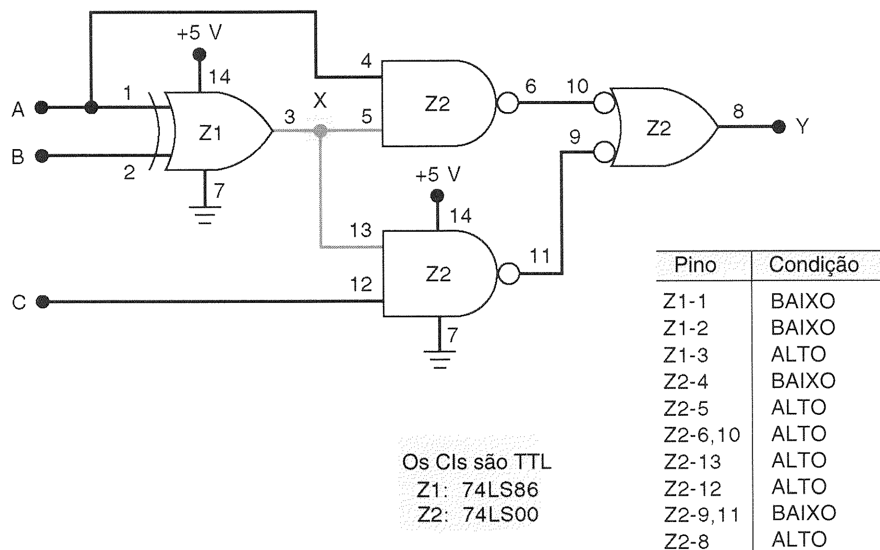
Considere o circuito da Fig. 4-39. A saída Y deveria estar em ALTO para as seguintes condições:

1. $A = 1$ e $B = 0$, independentemente do nível de C.
2. $A = 0$, $B = 1$ e $C = 1$.

Você mesmo pode verificar essas condições.

Quando o circuito é testado, o estudante observa que a saída Y vai para ALTO sempre que A ou C estão em ALTO, independentemente do nível de B. Ele faz as medições com uma ponta de prova para o caso em que $A = B = 0$ e $C = 1$ e anota as indicações, conforme pode ser visto na Fig. 4-39.

Examine os níveis observados e relacione as possíveis causas do mau funcionamento. Depois disso, desenvolva um procedimento com etapas para determinar qual é a falha de fato.



Pino	Condição
Z1-1	BAIXO
Z1-2	BAIXO
Z1-3	ALTO
Z2-4	BAIXO
Z2-5	ALTO
Z2-6,10	ALTO
Z2-13	ALTO
Z2-12	ALTO
Z2-9,11	BAIXO
Z2-8	ALTO

Fig. 4-39 Exemplo 4-28.

Solução

Todas as saídas das portas NAND estão corretas em função dos níveis presentes em suas entradas. Entretanto, a porta EX-OR deveria ter produzido um nível BAIXO no pino de saída 3, uma vez que ambas as entradas estão em BAIXO. Aparentemente, Z1-3 está sempre em ALTO, mesmo quando suas entradas indicam que ela deveria estar em BAIXO. As possíveis causas para que isto aconteça estão listadas a seguir:

1. Um componente interno danificado em Z1 que impede que a saída vá para BAIXO.
2. Um curto externo entre V_{cc} e qualquer ponto ao longo do caminho de condução para o ponto *X*.
3. Um curto interno entre Z1-3 e V_{cc}.
4. Um curto interno entre Z2-5 e V_{cc}.
5. Um curto interno entre Z2-13 e V_{cc}.

Todas essas alternativas, com exceção da primeira, vão colocar em curto o ponto *X* (e qualquer pino conectado a ele) e V_{cc}.

O procedimento a seguir pode ser usado para isolar a falha. Este não é o único que pode ser usado, e, como afirmamos anteriormente, o procedimento a ser usado em cada caso depende da pessoa que está fazendo a depuração e dos equipamentos de teste disponíveis.

1. Verifique os níveis de tensão dos pinos de V_{cc} e terra de Z1. Embora seja improvável que a ausência de um desses níveis pudesse fazer com que Z1-3 ficasse em ALTO, é uma boa prática fazer essa verificação em qualquer CI que esteja produzindo uma saída incorreta.
2. Desligue a fonte de alimentação do circuito e use um ohmímetro para verificar se existe um curto (uma resistência menor que 1 Ω) entre o ponto *X* e qualquer ponto conectado a V_{cc} (como Z1-14 ou Z2-14). Se não houver indicação de um curto, as últimas quatro possibilidades da nossa lista podem ser eliminadas. Isto significa que Z1 está danificado e deve ser trocado.
3. Se na etapa 2 existir um curto entre o ponto *X* e V_{cc}, faça uma inspeção visual na placa do circuito e procure pontes de solda, resíduos de cobre não-corroído, fios desencapados em contato e qualquer outra possível causa de curto com V_{cc}. Um lugar provável para encontrar uma ponte de solda é entre os pinos adjacentes 13 e 14 de Z2, pois o pino 14 é conectado ao V_{cc} e o pino 13 ao ponto *X*. Se um curto externo for encontrado, remova-o e faça uma verificação com um ohmímetro para determinar se o ponto *X* não está mais em curto com V_{cc}.
4. Se a etapa 3 não revelar nenhum curto externo, as possíveis causas que nos restam são curtos internos entre V_{cc} e Z1-3, Z2-13 ou Z2-5. Um destes está colocando o ponto *X* em curto com V_{cc}.

Para determinar qual desses CIs é o culpado, você deve desconectar cada um deles do ponto *X*, *um de cada vez*, e verificar se o curto com V_{cc} permanece após cada desconexão, pois quando o pino que está internamente em curto com V_{cc} for desconectado o ponto *X* não estará mais em curto com V_{cc}.

O processo de desconectar cada um dos pinos suspeitos do ponto *X* pode ser fácil ou difícil, dependendo de como o circuito está montado. Se os CIs estão em soquetes, tudo o que você tem a fazer é retirar o CI do soquete, dobrar o pino suspeito para fora e recolocar o CI no soquete. Se os CIs estão soldados na placa de circuito impresso, você terá que cortar a trilha na qual o pino está conectado, ou o próprio pino. Após o teste ter sido feito, você terá que fazer reparos na trilha ou no pino, conforme o caso.

Existe uma técnica de depuração que torna desnecessário dobrar pinos ou cortar trilhas quando estamos tentando isolar um curto. Ela envolve a utilização de um equipamento chamado *rastreador de corrente*. Este instrumento é capaz de rastrear o fluxo de corrente que atravessa um curto-circuito, através da mudança do fluxo magnético em torno

do condutor onde se verifica o curto. Vamos examinar isto em detalhe no Cap. 8.

O Exemplo 4-28, apesar de ser bastante simples, mostra os tipos de raciocínio que alguém que faz depuração de circuitos (engenheiro, técnico ou estudante) deve empregar de modo a isolar a falha do circuito. Você terá a oportunidade de desenvolver suas habilidades de depuração de circuitos resolvendo os Problemas 4-34 a 4-44.

4-14 LÓGICA PROGRAMÁVEL*

Como foi visto anteriormente, a última etapa no projeto de um circuito lógico é reunir os CIs necessários e fazer as conexões apropriadas, de modo que as saídas deste circuito sejam as funções lógicas das entradas que desejamos. Uma vez feito o projeto, este circuito pode ser testado. Você provavelmente já fez isso muitas vezes no laboratório.

Mais adiante neste livro gastaremos bastante tempo aprendendo uma outra maneira de se implementar funções lógicas, que é bastante diferente desta que estamos aprendendo agora. Ela utiliza algo a que chamamos *lógica programável* e é especialmente útil na implementação de circuitos mais complexos que contêm dezenas ou centenas de portas lógicas. Discutiremos os detalhes da lógica programável mais tarde, mas o conceito básico será introduzido aqui com o auxílio da Fig. 4-40. O bloco retangular representa conceitualmente um exemplo de um **dispositivo de lógica programável** (**PLD** — *programmable logic device*), que é um circuito integrado que contém um arranjo particular de portas lógicas. Existem muitos tipos de PLDs; todos eles contêm muito mais do que as poucas portas lógicas que vemos na Fig. 4-40. Entretanto, usaremos este exemplo simples para mostrar a idéia básica de toda lógica programável.

Podemos reconhecer a lógica utilizada nesse PLD simplificado como uma estrutura do tipo soma-de-produtos, onde as saídas das portas AND são conectadas a uma porta OR. A saída X será uma função do tipo soma-de-produtos das entradas A e B. A função de saída implementada dependerá de quais saídas das portas AND são conectadas às entradas da porta OR. Na figura, todas as saídas das portas AND são conectadas às entradas da porta OR através dos elos de ligação 1, 2, 3 e 4. Estes elos podem ficar intactos, como mostrado na figura, ou podem ser seletivamente abertos, para desconectar a saída correspondente da entrada da porta OR. Por exemplo, se os elos 1, 2 e 3 fossem abertos, apenas a porta AND 4 estaria conectada, e a saída da porta OR seria $X = AB$. Se os elos 1 e 4 fossem abertos, a saída seria $X = A\overline{B} + \overline{A}B$. O circuito interno é tal que um elo aberto produz um nível BAIXO na sua entrada da porta OR.

Desse modo, podemos implementar qualquer expressão do tipo soma-de-produtos de duas variáveis abrindo os elos apropriados. O chip PLD vem com todos os seus elos intactos, e todos eles estão dentro do chip. Como isto é feito? Bem, descobriremos mais tarde. Por enquanto, vamos dizer que o PLD tem entradas de programação (representadas pela grande seta no diagrama) que podemos usar para, de algum modo, abrir determinados elos de modo a implementar a função lógica em particular que queremos. Este procedimento é chamado de *programação* do PLD. As entradas de programação são usadas somente durante o processo de programação, para configurar as conexões internas do chip. Uma vez feito isto, o PLD foi *programado* para realizar a operação lógica sobre suas entradas A e B e está pronto para ser utilizado com esse objetivo.

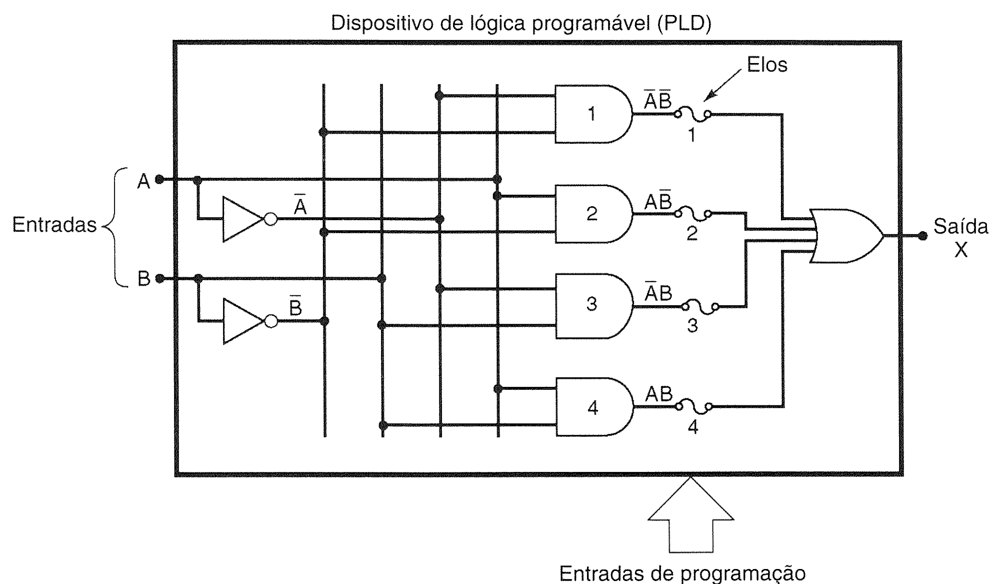


Fig. 4-40 Exemplo simplificado de um dispositivo de lógica programável.

*Esta seção pode ser omitida sem que haja perda de continuidade.

A estrutura mostrada na Fig. 4-40 é apenas um dos muitos tipos de estruturas de PLD que estudaremos nos Caps. 11 e 12. É relativamente fácil ver como essa estrutura AND-OR pode ser expandida para acomodar um número maior de entradas. Por exemplo, para três entradas, o PLD teria três INVERSORes, oito portas AND (uma para cada produto AND possível das três variáveis), oito elos e uma porta OR de 8 entradas.

Questões de Revisão

1. Qual seria a função de saída do PLD se os elos 1 e 2 na Fig. 4-40 fossem abertos?
2. Qual seria a função se todos os elos permanecessem intactos?
3. Descreva os componentes de um PLD que está estruturado como visto na Fig. 4-40 e que possui *quatro* entradas.

RESUMO

1. As duas formas gerais para as expressões lógicas são soma-de-produtos e produto-de-somas.
2. Um dos métodos de projeto de circuitos lógicos combinacionais é constituído das seguintes etapas: (1) construir sua tabela-verdade; (2) converter a tabela-verdade em uma expressão do tipo soma-de-produtos; (3) simplificar esta expressão utilizando álgebra booleana ou mapa de Karnaugh; (4) implementação da expressão final obtida.
3. O mapa de Karnaugh é um método gráfico de representar a tabela-verdade do circuito e gerar a expressão simplificada para a saída do mesmo.
4. Um circuito EX-OR tem como expressão $x = A\overline{B} + \overline{A}B$. Sua saída será ALTA apenas quando as entradas A e B estiverem em níveis lógicos opostos.
5. Um circuito EX-NOR tem como expressão $x = \overline{A}\overline{B} + AB$. Sua saída será ALTA apenas quando as entradas A e B estiverem em um mesmo nível lógico.
6. Cada uma das portas básicas (AND, OR, NAND e NOR) pode ser usada para habilitar/desabilitar a passagem de um sinal de entrada para a sua saída.
7. As principais famílias de CIs são a TTL e a CMOS. Os CIs digitais disponíveis implementam uma grande variedade de funções lógicas, desde as mais simples (poucas portas por chip) até as mais complexas.
8. Para realizar a depuração básica de um circuito, você necessita, pelo menos, compreender o seu funcionamento, conhecer os tipos de falhas possíveis, um diagrama esquemático completo e uma ponta de prova lógica.
9. Um dispositivo lógico programável (PLD) é um CI que contém um grande número de portas lógicas cujas interconexões podem ser programadas pelo usuário para gerar a relação lógica desejada entre as entradas e as saídas.

TERMOS IMPORTANTES

soma-de-produtos
produto-de-somas
mapa de Karnaugh (mapa K)
agrupamento
condição *don't care*
exclusive-OR (EX-OR)
exclusive-NOR (EX-NOR)
geração e verificação de paridade
habilitado/desabilitado
Dual-In-Line Package (DIP)
SSI, MSI, LSI, VLSI, ULSI, GSI
Transistor-Transistor Logic (TTL)
Complementary Metal Oxide Semiconductor (CMOS)
indeterminado(a)(s)
flutuação
ponta de prova lógica
dispositivo de lógica programável (PLD)

PROBLEMAS

SEÇÕES 4-2 E 4-3

4-1. Simplifique as expressões a seguir usando a álgebra booleana.

(a) $x = ABC + \overline{A}C$
(b) $y = (Q + R)(\overline{Q} + \overline{R})$
(c) $w = ABC + A\overline{B}C + \overline{A}$
(d) $q = \overline{RST}\,(\overline{R + S + T})$
(e) $x = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}BC + ABC + A\overline{B}\,\overline{C} + A\overline{B}C$
(f) $z = (B + \overline{C})(\overline{B} + C) + \overline{\overline{A} + B + \overline{C}}$
(g) $y = \overline{(C + D)} + \overline{A}C\overline{D} + A\overline{B}\,\overline{C} + \overline{A}\,\overline{B}CD + AC\overline{D}$

4-2. Simplifique o circuito da Fig. 4-41 usando a álgebra booleana.

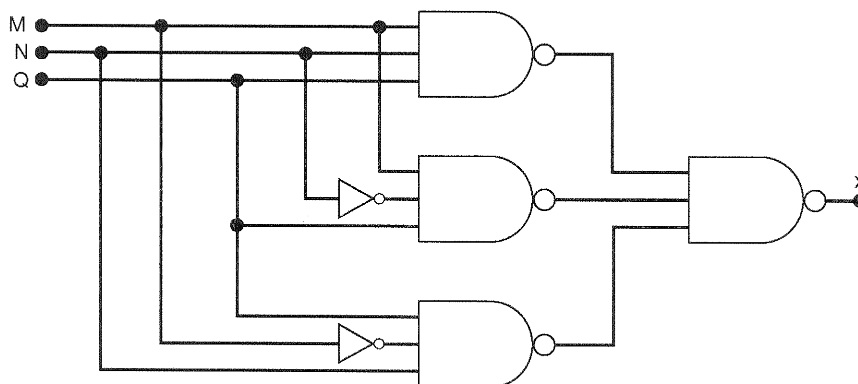


Fig. 4-41 Problemas 4-2 e 4-3.

4-3. Troque cada uma das portas do Problema 4-2 por uma porta NOR e simplifique o circuito usando a álgebra booleana.

SEÇÃO 4-4

D

4-4. Projete um circuito lógico que corresponde à tabela-verdade mostrada na Tabela 4-8.

TABELA 4-8

A	*B*	*C*	*x*
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

D

4-5. Projete um circuito lógico no qual a saída está em ALTO *apenas* quando a maioria das entradas *A*, *B* e *C* está em BAIXO.

D

4-6. Uma fábrica necessita de uma sirene para indicar o fim do expediente. Esta sirene deve ser tocada em uma das seguintes condições:

1. Já passa das cinco horas e todas as máquinas estão desligadas.

2. É sexta-feira, a produção do dia foi atingida e todas as máquinas estão desligadas.

Projete um circuito que controle a sirene. (*Sugestão*: Use quatro variáveis lógicas para representar as diversas condições. Por exemplo, a entrada *A* estará em ALTO somente quando já for mais de cinco horas.)

D

4-7. Um número binário de quatro bits é representado por $A_3A_2A_1A_0$ onde A_3, A_2, A_1 e A_0 representam cada um dos bits, sendo A_0 o LSB. Projete um circuito que produz uma saída em ALTO sempre que o número for maior do que 0010 e menor do que 1000.

D

4-8. A Fig. 4-42 mostra um diagrama de um circuito de alarme de automóvel usado para detectar algumas situações indesejáveis. As três chaves são usadas para indicar, respectivamente, o estado da porta do motorista, da ignição e dos faróis. Projete um circuito que tenha como entrada essas três chaves e ative o alarme em uma das seguintes condições:

- Os faróis estão acesos e a ignição está desligada.
- A porta do motorista está aberta e a ignição está ligada.

4-9. Implemente o circuito do Problema 4-4 usando apenas portas NAND.

4-10. Implemente o circuito do Problema 4-5 usando apenas portas NAND.

4-11. Implemente a expressão $z = \overline{D} + ABC + \overline{A}\overline{C}$, utilizando portas AND, OR e INVERSORes. A partir do circuito obtido, construa um outro usando somente portas NAND.

SEÇÃO 4-5

4-12. Simplifique a expressão do Problema 4-1(e) usando o mapa K.

4-13. Simplifique a expressão do Problema 4-1(g) usando o mapa K.

4-14. Simplifique a expressão do Problema 4-7 usando o mapa K.

4-15. Determine a expressão mínima para cada mapa K da Fig. 4-43. Para o mapa do item (a), preste atenção especialmente no passo 5.

(a)

	$\overline{C}\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\overline{B}$	1	1	1	1
$\overline{A}B$	1	1	0	0
AB	0	0	0	1
$A\overline{B}$	0	0	1	1

(b)

	$\overline{C}\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$
$\overline{A}\overline{B}$	1	0	1	1
$\overline{A}B$	1	0	0	1
AB	0	0	0	0
$A\overline{B}$	1	0	1	1

(c)

	$\overline{C}$	C
$\overline{A}\overline{B}$	1	1
$\overline{A}B$	0	0
AB	1	0
$A\overline{B}$	1	X

Fig. 4-43 Problema 4-15.

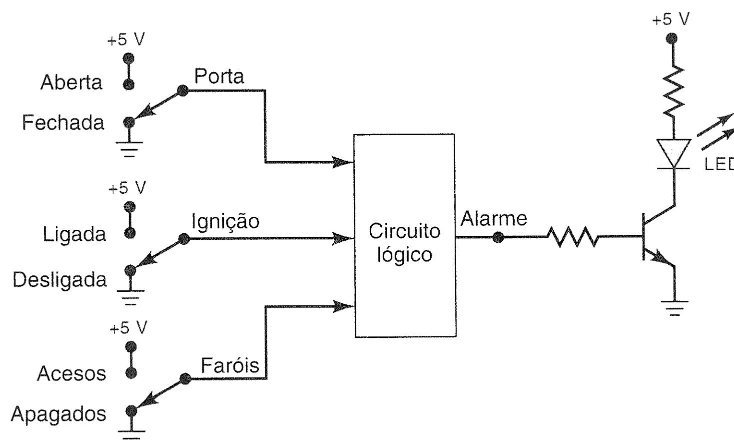


Fig. 4-42 Problema 4-8.

D

4-16. A Fig. 4-44 mostra um *contador BCD* que produz uma saída de quatro bits, que representa, em código BCD, o número de pulsos que foram aplicados à entrada do contador. Por exemplo, após quatro pulsos, a saída do contador é $DCBA = 0100_2 = 4_{10}$. O contador retorna a 0000 no décimo pulso e inicia a contagem novamente. Assim, as saídas $DCBA$ nunca representarão um número maior que $1001_2 = 9_{10}$. Projete um circuito que produza um nível ALTO quando a contagem for igual a 2, 3 ou 9. Use o mapa de Karnaugh e aproveite as condições *don't care*.

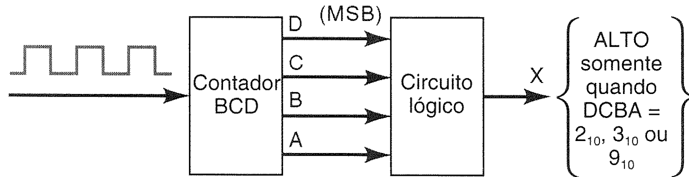


Fig. 4-44 Problema 4-16.

D

4-17. A Fig. 4-45 mostra quatro chaves que são parte de um circuito de controle de uma máquina copiadora. As chaves estão localizadas ao longo do caminho que o papel passa pela máquina. Cada uma das chaves está normalmente aberta, e quando o papel passa pela chave, ela é fechada. É impossível que as chaves SW1 e SW4 estejam fechadas ao mesmo tempo. Projete um circuito que produza uma saída em ALTO quando *duas ou mais* chaves estiverem fechadas ao mesmo tempo. Use o mapa de Karnaugh e aproveite as condições *don't care*.

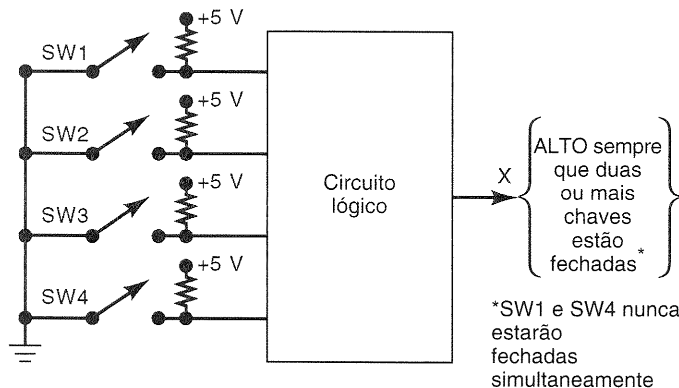


Fig. 4-45-Problema 4-17.

SEÇÃO 4-6

4-18. **(a)** Determine a forma de onda da saída do circuito da Fig. 4-46.

(b) Repita o item (a) para o caso em que B está permanentemente em BAIXO.

(c) Repita o item (a) para o caso em que B está permanentemente em ALTO.

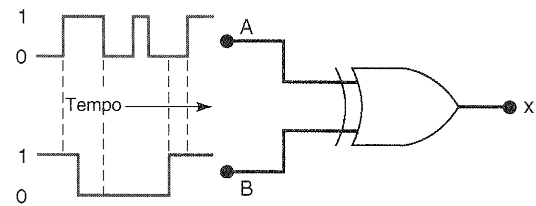


Fig. 4-46 Problema 4-18.

4-19. Determine as condições de entrada necessárias para fazer $x = 1$ na Fig. 4-47.

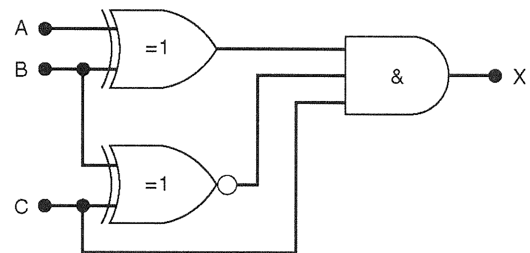


Fig. 4-47 Problema 4-19.

C, D

4-20. Projete um circuito com três entradas que produza uma saída em ALTO apenas quando todas as entradas estiverem em um mesmo nível. Use apenas portas EX-OR e uma outra porta de outro tipo.

C

4-21. A Fig. 4-48 representa um *detector de magnitude relativa*. Este circuito recebe dois números binários de três bits $x_2x_1x_0$ e $y_2y_1y_0$ e determina se os números são iguais, e, se não forem, indica qual é o maior. As três saídas estão definidas a seguir:

1. $M = 1$ apenas se os dois números forem iguais.
2. $N = 1$ apenas se $x_2x_1x_0$ for maior que $y_2y_1y_0$.
3. $P = 1$ apenas se $y_2y_1y_0$ for maior que $x_2x_1x_0$.

Projete o circuito para esse detector. Ele possui seis entradas e três saídas e portanto é complexo demais para usar uma tabela-verdade. A título de sugestão, estude o Exemplo 4-16. Ele pode lhe dar uma dica de como resolver este problema.

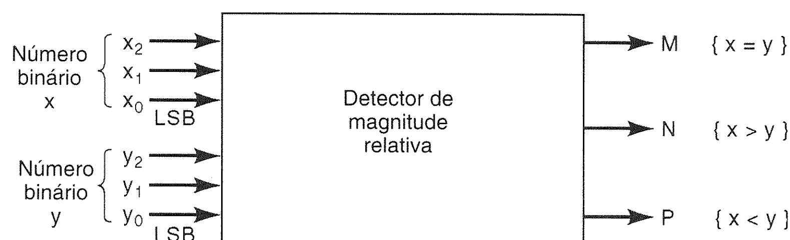


Fig. 4-48 Problema 4-21.

MAIS PROBLEMAS DE PROJETO

C, D

4-22. A Fig. 4-49 apresenta um circuito multiplicador que recebe dois números binários de dois bits x_1x_0 e y_1y_0 e produz como saída um número binário $z_3z_2z_1z_0$ que é igual ao produto aritmético dos números de entrada. Projete um circuito para o multiplicador. (*Sugestão*: O circuito possui quatro entradas e quatro saídas.)

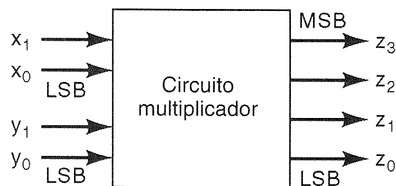


Fig. 4-49 Problema 4-22.

D

4-23. Um código BCD está sendo transmitido a um receptor remoto. Os bits são $A_3A_2A_1A_0$, onde A_3 é o MSB. Dentre os circuitos existentes no receptor existe um denominado *detector de erros BCD*. Este circuito verifica se o código recebido é um código BCD válido (isto é, ≤ 1001). Projete este circuito de modo a produzir um nível ALTO em uma situação de erro.

D

4-24. Projete um circuito cuja saída esteja em ALTO sempre que *A* e *B* estiverem em ALTO e enquanto *C* e *D* estiverem ambos em nível ALTO, ou ambos em BAIXO. Tente fazer este exercício sem utilizar uma tabela-verdade. Verifique o resultado construindo uma tabela-verdade para o seu circuito, e observe se ele obedece ao enunciado do problema.

D

4-25. Quatro grandes tanques em uma indústria química contêm diferentes líquidos que estão sendo aquecidos. Sensores de nível de líquido são utilizados para detectar se o nível do tanque *A* ou do tanque *B* sobe acima de um nível predeterminado. Sensores de temperatura existentes nos tanques *C* e *D* detectam se a temperatura de um desses tanques cai abaixo de um determinado limite. Suponha que as saídas dos sensores de nível de líquido *A* e *B* estarão em BAIXO quando o nível for satisfatório e estarão em ALTO quando o nível for muito alto. Além disso, as saídas dos sensores de temperatura *C* e *D* estarão em BAIXO quando a temperatura for satisfatória e estarão em ALTO quando a temperatura for muito BAIXA. Projete um circuito que detecte quando o nível no tanque *A* ou no *B* estiver muito alto, ao mesmo tempo em que a temperatura em um dos tanques *C* ou *D* estiver muito baixa.

C, D

4-26. A Fig. 4-50 mostra o cruzamento de uma rodovia com uma via de acesso. Sensores detectores de veículos são colocados ao longo das pistas *C* e *D* da rodovia e das pistas *A* e *B* da via de acesso. A saída desse tipo de sensor está em BAIXO quando não existe nenhum carro presente e está em ALTO quando um veículo está presente. Um sinal de trânsito colocado no cruzamento deve funcionar de acordo com a seguinte lógica:

1. O sinal da direção leste-oeste (L-O) deve estar verde quando *ambas* as pistas *C* e *D* estiverem ocupadas.
2. O sinal da direção (L-O) deve estar verde quando *ou C* ou *D* estiverem ocupadas mas *ambas* as pistas *A* e *B* não estiverem.
3. O sinal da direção norte-sul (N-S) deve estar verde quando *ambas* as pistas *A* e *B* estiverem ocupadas mas *ambas* as pistas *C* e *D* não estiverem.
4. O sinal da direção (N-S) deve estar verde quando *ou A* ou *B* estiverem ocupadas e enquanto *ambas* as pistas *C* e *D* estiverem vazias.
5. O sinal da direção (L-O) deve estar verde quando *não* houver nenhum veículo presente.

Utilizando as saídas dos sensores *A*, *B*, *C* e *D* como entradas, projete um circuito que controle esse sinal de trânsito. Devem existir duas saídas, N-S e L-O, que devem ir para ALTO quando o sinal correspondente tiver que estar *verde*. Simplifique esse circuito ao máximo, e mostre todas as etapas de simplificação.

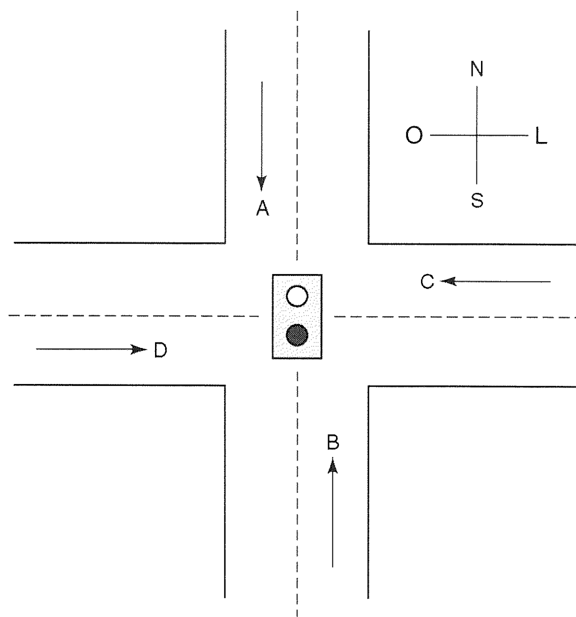


Fig. 4-50 Problema 4-26.

SEÇÃO 4-7

D

4-27. Projete novamente o verificador e gerador de paridade da Fig. 4-24 para operar com paridade ímpar. (*Sugestão*: Qual é a relação entre o bit de paridade, quando usamos paridade ímpar, e aquele que é usado quando usamos paridade par para um mesmo conjunto de bits de dados?)

D

4-28. Projete novamente o verificador e gerador de paridade da Fig. 4-24 para operar com oito bits de dados.

SEÇÃO 4-8

D

4-29. Projete um circuito que permitirá que o sinal de entrada *A* chegue até a saída somente quando a entrada de controle *B* estiver em BAIXO e enquanto a outra entrada de controle *C* estiver em ALTO. Caso isso não ocorra, a saída deve estar em BAIXO.

D

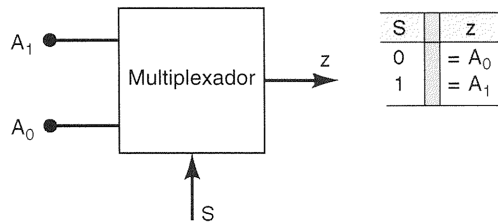
4-30. Projete um circuito que *desabilita* a passagem do sinal de entrada somente quando as entradas de controle *B*, *C* e *D* estiverem em ALTO. Quando a passagem estiver desabilitada, a saída deve estar em ALTO.

D

4-31. Projete um circuito que controla a passagem de um sinal *A* de acordo com os seguintes requisitos:

1. A saída X deve ser igual a A quando as entradas de controle B e C estiverem em um mesmo nível.
2. X deve permanecer em ALTO quando B e C estiverem em níveis diferentes.

D
4-32. Projete um circuito que possui dois sinais de entrada A_1 e A_0, uma entrada de controle S, e funciona conforme os requisitos mostrados na Fig. 4-51. Esse tipo de circuito é chamado de *multiplexador* (e será estudado no Cap. 9).



S	z
0	= A_0
1	= A_1

Fig. 4-51 Problema 4-32.

D
4-33. Use o mapa de Karnaugh para projetar um circuito que obedeça aos requisitos do Exemplo 4-16. Compare o circuito obtido com aquele que está na Fig. 4-22. Este exercício mostra que o mapa de Karnaugh não pode aproveitar-se das portas lógicas EX-OR e EX-NOR. O projetista deve ser capaz de determinar quando estas portas são aplicáveis.

SEÇÕES 4-9 A 4-13

T*
4-34. **(a)** Um técnico está testando um circuito lógico e verifica que a saída de um determinado INVERSOR está sempre em BAIXO, mesmo que sua entrada esteja pulsando. Enumere as possíveis razões deste mau funcionamento.
(b) Repita o item (a) para o caso em que a saída do INVERSOR está sempre em um nível indeterminado.

T
4-35. Os sinais mostrados na Fig. 4-52 são aplicados às entradas do circuito da Fig. 4-31. Suponha que existe um circuito aberto interno em Z1-4.
(a) O que a ponta de prova lógica indicaria em Z1-4?
(b) Qual seria a tensão contínua que você esperaria ler em um voltímetro colocado em Z1-4? (Lembre-se que todos os CIs são TTL.)
(c) Faça um esboço de como seriam os sinais $\overline{CLOCKOUT}$ e $\overline{SHIFTOUT}$.
(d) Em vez de um circuito aberto, suponha que os pinos 9 e 10 de Z2 estejam em curto internamente. Desenhe como seriam os sinais em Z2-10, $\overline{CLOCKOUT}$ e $\overline{SHIFTOUT}$.

T
4-36. Suponha que os CIs da Fig. 4-31 são CMOS. Descreva como a operação deste circuito seria afetada se houvesse um circuito aberto na ligação entre Z2-2 e Z2-10.

T
4-37. No Exemplo 4-27, relacionamos três possíveis falhas para a situação mostrada na Fig. 4-34. Que procedimento você utilizaria para determinar qual das possíveis falhas está realmente causando o problema?

*Lembre-se de que **T** significa que o exercício é de depuração [do inglês *troubleshooting*].

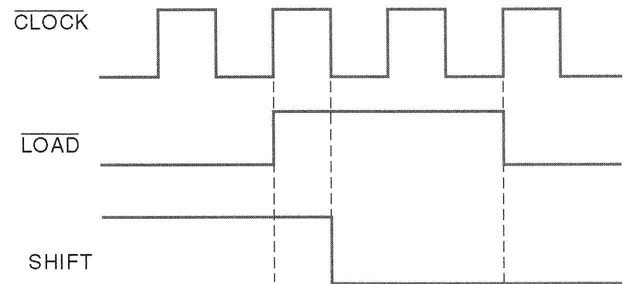


Fig. 4-52 Problema 4-35.

T
4-38. Veja o circuito da Fig. 4-36. Suponha que esses dispositivos são todos CMOS. Além disso, suponha que a indicação da ponta de prova lógica em Z2-3 é "indeterminado", em vez de pulsante. Relacione as possíveis falhas e escreva um procedimento que o ajude a determinar qual dessas falhas está provocando o problema.

T
4-39. Veja o circuito da Fig. 4-36. Recorde que a saída Y deve estar em ALTO para qualquer uma das seguintes condições:
1. $A = 1$, $B = 0$, independentemente de C.
2. $A = 0$, $B = 1$ e $C = 1$.
Durante o teste do circuito, o técnico observa que Y vai para ALTO somente para a primeira condição, mas permanece em BAIXO para todas as outras. Considere a seguinte lista de possíveis falhas. Para cada uma delas, escreva "sim" ou "não" para indicar se essa falha pode ou não ser a causa do problema. Explique o seu raciocínio para cada item marcado com um "não".
(a) Um curto interno entre terra e Z2-13.
(b) Um circuito aberto na ligação para Z2-13.
(c) Um curto interno entre V_{cc} e Z2-11.
(d) Um circuito aberto na ligação de V_{cc} para Z2.
(e) Um circuito aberto interno em Z2-9.
(f) Um circuito aberto na ligação entre Z2-11 e Z2-9.
(g) Uma ponte de solda entre os pinos 6 e 7 de Z2.

T
4-40. Desenvolva um procedimento para isolar a falha que está causando o mau funcionamento descrito no Problema 4-39.

T
4-41. Suponha que as portas lógicas da Fig. 4-39 são todas CMOS. Quando um técnico testa o circuito, ele percebe que o mesmo opera corretamente, exceto nas seguintes condições:
1. $A = 1$, $B = 0$, $C = 0$.
2. $A = 0$, $B = 1$, $C = 1$.
Para essas condições, a ponta de prova indica níveis indeterminados em Z2-6, Z2-11 e Z2-8. Você seria capaz de dizer qual a causa provável do mau funcionamento? Explique o seu raciocínio.

T
4-42. A Fig. 4-53 é um circuito lógico combinacional que ativa o alarme do carro sempre que os assentos do motorista e/ou do passageiro estão ocupados mas o cinto de segurança não está colocado quando o carro é ligado. Os sinais *DRIV* e *PASS* são ativos em ALTO e indicam a presença do motorista e do passageiro, respectivamente. Estes sinais são fornecidos por chaves ativadas por pressão colocadas nos assentos. O sinal *IGN* é ativo em ALTO quando a chave de ignição está ligada. O sinal $\overline{BELTD}$ é ativo em BAIXO e indica que o cinto de segurança do motorista *não* está colocado. O sinal $\overline{BELTP}$ é o sinal referente ao cinto do passageiro. O alarme será ativado (BAIXO) sempre que o carro for ligado e um dos bancos dianteiros estiver ocupado e seu cinto não estiver colocado.

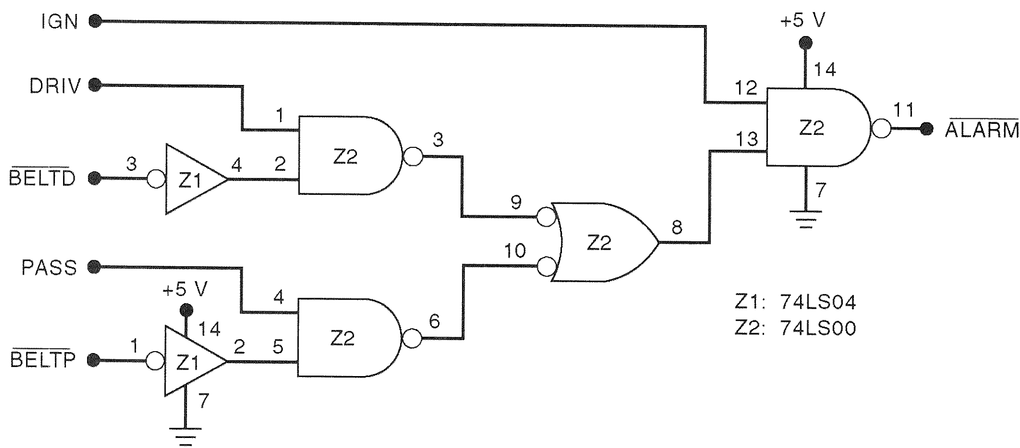


Fig. 4-53 Problemas 4-42, 4-43 e 4-44.

(a) Verifique que o circuito funciona conforme a descrição dada.

(b) Descreva como esse sistema de alarme iria operar se Z1-2 estivesse internamente em curto com a terra.

(c) Descreva como esse circuito iria operar se existisse um circuito aberto na ligação entre Z2-6 e Z2-10.

T

4-43. Suponha que o sistema da Fig. 4-53 está funcionando de tal modo que o alarme é ativado assim que o motorista ou o passageiro estejam sentados e o carro seja ligado, sem levar em conta se os cintos estão colocados ou não. Quais são as possíveis falhas que podem estar ocorrendo? Como você faria para determinar a falha que está causando esse problema?

T

4-44. Suponha que o sistema da Fig. 4-53 está funcionando de tal modo que o alarme é ativado tão logo a ignição seja ligada, não importando o estado das outras entradas. Relacione as possíveis falhas e escreva um procedimento para isolar a falha que está causando o problema.

SEÇÃO 4-14

C

4-45. (a) Modifique a estrutura do PLD da Fig. 4-40 de modo que ela possa receber três entradas.

(b) Usando esse PLD de três entradas, mostre como implementar o circuito do Exemplo 4-7. Observe que não é necessário simplificar a expressão lógica da saída para fazer isso.

(c) O circuito do Exemplo 4-7 foi implementado usando um CI NAND quádruplo [Fig. 4-9(a)]. Compare o número de conexões *externas* entre CIs dessa implementação com a outra que utiliza um PLD.

QUESTÃO DE FIXAÇÃO

4-46. Defina cada um dos seguintes termos.

(a) Mapa de Karnaugh

(b) Forma de soma-de-produtos

(c) Gerador de paridade

(d) Octeto

(e) Circuito habilitador

(f) Estado *don't care*

(g) Entrada em flutuação

(h) Nível de tensão indeterminado

APLICAÇÕES EM MICROCOMPUTADORES

C

4-47. Em um microcomputador, a unidade microprocessadora (MPU — *MicroProcessor Unit*) está sempre se comunicando com um dos seguintes dispositivos: (1) memória de acesso aleatório (RAM — *random access memory*), que armazena programas e dados que podem ser prontamente modificados; (2) memória apenas de leitura (ROM — *read only memory*), que armazena programas e dados que nunca são modificados; (3) dispositivos externos de entrada e saída (E/S), tais como: teclados, monitores de vídeo, impressoras e unidades de disco. Quando está executando um programa, a MPU gera o endereço que seleciona o tipo de dispositivo (RAM, ROM ou E/S) com o qual ela quer se comunicar. A Fig. 4-54 mostra um esquema típico em que a MPU gera oito bits de endereço, de A_{15} até A_8. Na verdade, a MPU gera um endereço de dezesseis bits; entretanto, os bits de ordem mais baixa de A_7 até A_0 não são utilizados nos processos de seleção do dispositivo. O endereço é fornecido como entrada de um circuito que, então, gera os seguintes sinais de seleção de dispositivos: $\overline{RAM}$, $\overline{ROM}$ e $\overline{E/S}$.

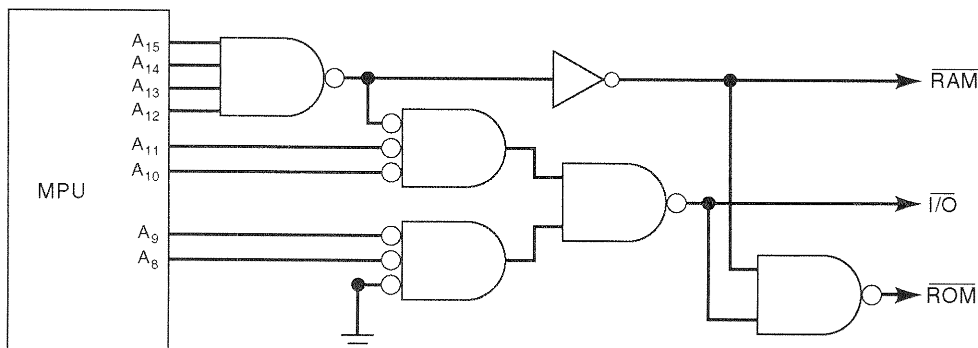


Fig. 4-54

Analise o circuito e responda:

(a) A faixa de endereços de A_{15} até A_8 que irá ativar o sinal $\overline{RAM}$.

(b) A faixa de endereços que irá ativar o sinal $\overline{E/S}$.

(c) A faixa de endereços que irá ativar o sinal $\overline{ROM}$.

Escreva os endereços em binário e em hexadecimal. Por exemplo, a resposta do item (a) é $A_{15} - A_8$: 00000000_2 até $11101111_2 = 00_{16}$ até EF_{16}.

C, D

4-48. Em alguns microcomputadores, a MPU pode ser *desabilitada* por curtos períodos de tempo, enquanto um outro dispositivo controla a RAM, ROM e E/S. Durante esse intervalo, um sinal especial de controle, $\overline{DMA}$, é ativado pela MPU, e é usado para desabilitar (desativar) a lógica de seleção de dispositivo, de modo que $\overline{\text{RAM}}$, $\overline{\text{ROM}}$ e $\overline{\text{E/S}}$ estão todos em seus estados inativos. Modifique o circuito da Fig. 4-54 para que $\overline{\text{RAM}}$, $\overline{\text{ROM}}$ e $\overline{\text{E/S}}$ sejam desativados sempre que $\overline{DMA}$ estiver ativo, independentemente do endereço.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 4-1

1. Somente (a)
2. Somente (c)

SEÇÃO 4-3

1. A expressão (b) não está na forma de soma-de-produtos, porque o sinal de inversão está sobre C e D (como por exemplo no termo $A\overline{CD}$). A expressão (c) não está na forma de soma-de-produtos, por causa do termo $(M + \overline{N})P$

3. $x = \overline{A} + \overline{B} + \overline{C}$

SEÇÃO 4-4

1. $x = \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}C\overline{D} + \overline{A}BC\overline{D}$
2. Oito

SEÇÃO 4-5

1. $x = AB + AC + BC$
2. $x = A + BCD$
4. Uma condição de entrada para qual não existe uma condição de saída especificada

SEÇÃO 4-6

2. A saída está permanentemente em BAIXO
3. Não. A porta EX-OR disponível pode ser usada como um INVERSOR se conectarmos uma das entradas a um nível ALTO constante (veja o Exemplo 4-15)

SEÇÃO 4-8

1. $x = \overline{A(B \oplus C)}$
2. OR, NAND
3. NAND, NOR

SEÇÃO 4-9

1. DIP
2. SSI, MSI, LSI, VLSI, ULSI e GSI
3. Verdadeiro
4. Verdadeiro
5. famílias 40, 74AC, 74ACT
6. 0 a 0,8 V; 2,0 a 5,0 V
7. 0 a 1,5 V; 3,5 a 5,0 V
8. Como se as entradas estivessem em ALTO
9. Imprevisível; pode superaquecer e ser destruído
10. 74HCT e 74ACT

SEÇÃO 4-11

1. Entradas ou saídas em aberto; entradas ou saídas em curto com V_{CC}; entradas ou saídas em curto com a terra; pinos em curto entre si; circuito internamente danificado
2. Pinos em curto entre si
3. Para TTL, nível BAIXO; para CMOS, indeterminado

SEÇÃO 4-12

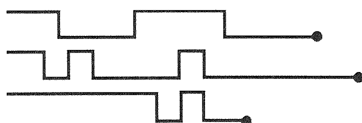
1. Circuito aberto em linhas de sinal; linhas de sinal em curto; fonte de alimentação com defeito; carregamento da saída
2. Fios partidos; soldas malfeitas; fissuras ou cortes na placa de circuito impresso; pinos de CI tortos ou amassados; soquetes defeituosos
3. CIs funcionando de modo incorreto ou simplesmente não funcionando
4. Nível lógico indeterminado

SEÇÃO 4-14

1. $x = B$
2. $x = 1$
3. Quatro INVERSORes, 16 portas AND, 16 elos e uma OR de 16 entradas

CAPÍTULO 5

Flip-Flops e Dispositivos Correlatos



SUMÁRIO

OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Construir um flip-flop latch com portas NAND, ou NOR, e analisar seu funcionamento.
- Eliminar os efeitos da trepidação de contato de uma chave mecânica utilizando um circuito latch.
- Descrever a diferença entre sistemas síncronos e assíncronos.
- Entender os diversos tipos de flip-flops disparados por transição tais como: o J-K, o D e o S-C.
- Analisar e aplicar os parâmetros de temporização dos flip-flops especificados pelos fabricantes.
- Descrever um circuito direcionador de pulsos e um circuito detector de transição.
- Compreender as principais diferenças entre as transferências de dados seriais e paralelas.
- Desenhar as formas de onda das saídas de diversos tipos de flip-flops, em resposta a um conjunto de sinais de entrada.
- Explicar os vários símbolos IEEE/ANSI para flip-flops.
- Utilizar diagramas de transição de estado para descrever a operação de contadores.
- Relacionar várias aplicações de flip-flops.
- Usar flip-flops em circuitos de sincronização.
- Conectar registradores de deslocamento como circuitos para transferências de dados.
- Empregar flip-flops em circuitos para divisão de freqüência e em contadores.
- Compreender as características típicas dos Schmitt triggers.
- Aplicar dois tipos diferentes de monoestáveis no projeto de circuitos.
- Projetar um oscilador utilizando um temporizador 555.
- Reconhecer e prever os efeitos que diferentes atrasos no sinal de clock provocam em circuitos síncronos.
- Depurar vários tipos de circuitos com flip-flops.

INTRODUÇÃO

Os circuitos lógicos estudados até agora são circuitos combinacionais, nos quais as saídas, em qualquer instante de tempo, dependem dos níveis presentes nas entradas no instante considerado. Quaisquer condições de entrada anteriores não têm efeito algum nas saídas atuais, porque os circuitos lógicos combinacionais não têm memória. A maioria dos sistemas digitais é composta tanto de circuitos combinacionais como de elementos de memória.

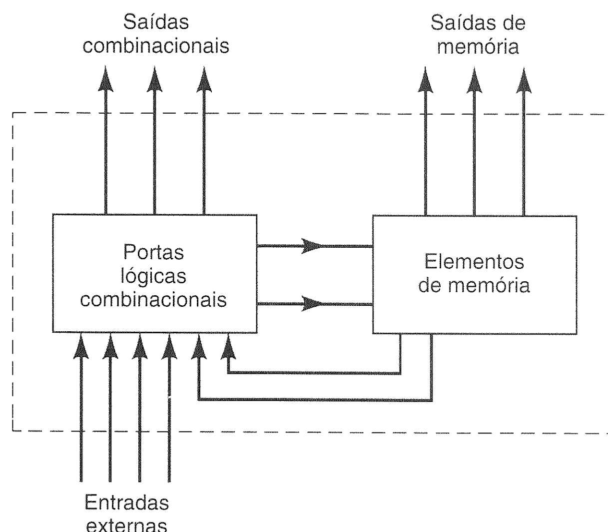


Fig. 5-1 Diagrama geral de um sistema digital.

A Fig. 5-1 mostra um diagrama de blocos de um sistema digital geral que reúne portas lógicas combinacionais com dispositivos de memória. A parte combinacional aceita sinais lógicos de entradas externas e das saídas dos elementos de memória. O circuito combinacional opera sobre estas entradas para produzir várias saídas, algumas das quais são usadas para determinar os valores binários a serem armazenados nos elementos de memória. As saídas de alguns elementos de memória, por outro lado, vão para as entradas das portas lógicas dos circuitos combinacionais. Este processo indica que as saídas externas de um sistema digital são uma função das entradas externas e das informações armazenadas nos seus elementos de memória.

O elemento de memória mais importante é o **flip-flop**, que é feito de uma configuração de portas lógicas. Embora uma porta lógica, por si só, não tenha capacidade de armazenamento, várias portas podem ser conectadas de modo a permitir que a informação seja armazenada. Muitas interconexões diferentes de portas são usadas para produzir flip-flops (abreviado como FFs).

A Fig. 5-2(a) mostra o símbolo utilizado para um flip-flop genérico. Ele possui duas saídas, identificadas como Q e $\overline{Q}$, que são opostas entre si. $Q/\overline{Q}$ são as designações mais comuns usadas para as saídas dos FFs. Por vezes, utilizaremos outras designações, tais como $X/\overline{X}$ e $A/\overline{A}$, por conveniência, na identificação de FFs diferentes em um circuito lógico.

A saída Q é chamada de saída *normal* do FF, e $\overline{Q}$ é a saída *invertida* (ou *barrada*) do FF. Sempre que nos referimos ao estado de um FF, estamos nos referindo ao estado de sua saída normal (Q). Fica subentendido que sua saída barrada ($\overline{Q}$) está no estado oposto. Por exemplo, se dissermos que um FF está no estado ALTO (1), significa que a saída $Q = 1$; se dissermos que um FF está no estado BAIXO (0), significa que a saída $Q = 0$. Obviamente, o estado de $\overline{Q}$ é sempre o inverso de Q.

Os dois estados possíveis de operação para um FF estão resumidos na Fig. 5-2(b). Note que o estado ALTO ou 1 ($Q = 1/\overline{Q} = 0$) também é chamado de estado **SET**. Sempre que as entradas de um FF fazem sua saída ir para o estado $Q =$

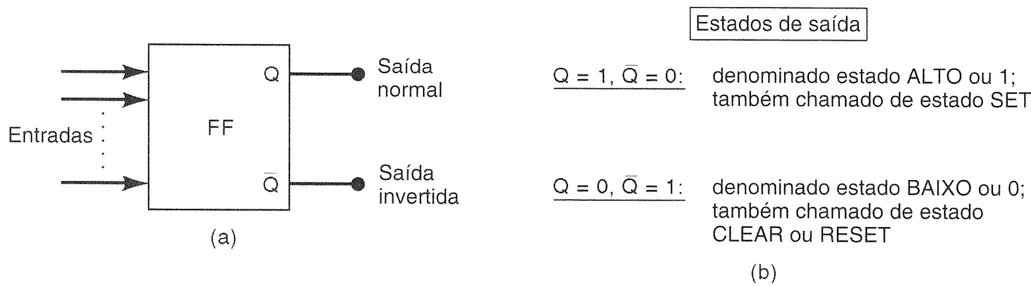


Fig. 5-2 Símbolo de um flip-flop genérico e definição dos dois estados de saída possíveis.

1, denominamos isto de *setar** o FF; o FF foi setado. Analogamente, o estado BAIXO ou 0 ($Q = 0/\overline{Q} = 1$) também é chamado de estado **CLEAR** ou **RESET**. Sempre que as entradas de um FF fazem sua saída ir para o estado $Q = 0$, denominamos isto de *limpar* ou *ressetar*** o FF; o FF foi limpo (ressetado). Conforme estudaremos, muitos FFs têm uma entrada **SET** e/ou uma entrada **CLEAR (RESET)** que são usadas para colocar o FF em um determinado estado de saída.

De acordo com o símbolo na Fig. 5-2(a), um FF pode ter uma ou mais entradas. Estas entradas são utilizadas para causar o chaveamento do FF entre seus possíveis estados de saída. Vamos descobrir que a maioria das entradas do FF necessita apenas ser momentaneamente ativada (pulsada) de modo a provocar uma mudança no estado de saída do FF, e a saída permanecerá neste novo estado mesmo após o pulso de entrada terminar. Esta é a propriedade de *memória* do FF.

O flip-flop é conhecido por outros nomes, incluindo *latch* e *multivibrador biestável*. O termo *latch* é usado para certos tipos de flip-flops que descreveremos. O termo *multivibrador biestável* é o nome mais técnico para um flip-flop, mas é muito complexo para ser utilizado regularmente.

5-1 LATCH COM PORTAS NAND

O circuito de FF mais básico pode ser construído com duas portas NAND ou com duas portas NOR. A versão com NAND, chamada de **latch com portas NAND** ou simplesmente **latch**, é mostrada na Fig. 5-3(a). As duas portas NAND são interligadas de modo cruzado, sendo que a saída da NAND-1 é conectada a uma das entradas da NAND-2, e vice-versa. As saídas das portas, identificadas como Q e $\overline{Q}$, respectivamente, são as saídas do latch. Sob condições normais, elas sempre são o inverso uma da outra. Existem duas entradas do latch: a entrada SET é a que *seta* Q para o estado 1; a entrada CLEAR é a que *limpa* Q para o estado 0.

Ambas as entradas SET e CLEAR estão normalmente em estado ALTO, e uma delas é pulsada em BAIXO sempre que se deseja alterar as saídas do latch. Vamos começar nossa análise mostrando que existem dois estados de saída similares quando SET = CLEAR = 1. Uma possibilidade é apresentada na Fig. 5-3(a), onde $Q = 0$ e $\overline{Q} = 1$. Com $Q = 0$, as entradas da NAND-2 são 0 e 1, o que produz $\overline{Q} = 1$. O nível 1 de $\overline{Q}$ faz com que NAND-1 tenha nível ALTO em ambas as entradas, o que resulta em 0 na saída Q. Com efeito, o que temos é um nível BAIXO na saída de NAND-1 produzindo um nível ALTO na saída de NAND-2, que por sua vez mantém a saída de NAND-1 em BAIXO.

A segunda possibilidade é mostrada na Fig. 5-3(b), onde $Q = 1$ e $\overline{Q} = 0$. O nível ALTO da porta NAND-1 produz um nível BAIXO na saída da NAND-2, o que por sua vez mantém a saída da NAND-1 em ALTO. Portanto, existem dois estados de saída possíveis quando SET = CLEAR = 1, e, como veremos em breve, o estado real depende do que ocorreu previamente com as entradas.

Setando o Latch (FF)

Vamos investigar agora o que acontece quando a entrada SET é momentaneamente pulsada em BAIXO, enquanto o CLEAR é mantido em ALTO. A Fig. 5-4(a) mostra o que acontece quando $Q = 0$ antes da ocorrência do pulso. Quando SET é pulsado BAIXO em t_0, Q vai para ALTO, e este nível ALTO força $\overline{Q}$ a ir para BAIXO, de modo que agora NAND-1 tem duas entradas em BAIXO. Logo, quando SET retorna para o estado 1 em t_1, a saída da NAND-1 *permanece* em ALTO, o que por sua vez mantém a saída da NAND-2 em BAIXO.

A Fig. 5-4(b) ilustra o que ocorre quando $Q = 1$ e $\overline{Q} = 0$ antes da aplicação do pulso em SET. Visto que a saída $\overline{Q} = 0$ já está mantendo a saída da NAND-1 em ALTO, o pulso BAIXO em SET não altera nada. Assim, quando SET retorna para ALTO, a saída do latch ainda está com $Q = 1$ e $\overline{Q} = 0$.

Pode-se resumir a Fig. 5-4 considerando-se que um pulso BAIXO na entrada SET sempre leva o latch para o estado onde $Q = 1$. Esta operação é denominada *setar* o latch ou FF.

Limpando o Latch (FF)

Vamos considerar agora o que ocorre quando a entrada CLEAR é pulsada em BAIXO enquanto SET é mantido em ALTO. A Fig. 5-5(a) mostra o que acontece quando $Q = 0$ e $\overline{Q} = 1$ antes da aplicação do pulso. Como $Q = 0$ já está mantendo a saída da NAND-2 em ALTO, o pulso em BAIXO no CLEAR não tem efeito algum. Quando CLEAR retornar para ALTO, as saídas do latch ainda serão $Q = 0$ e $\overline{Q} = 1$.

*Palavra de origem inglesa, comumente usada na área digital, que significa colocar a saída do FF igual a 1. (*N. do T.*)
**Palavra de origem inglesa, comumente usada na área digital, que significa colocar a saída do FF igual a 0. (*N. do T.*)

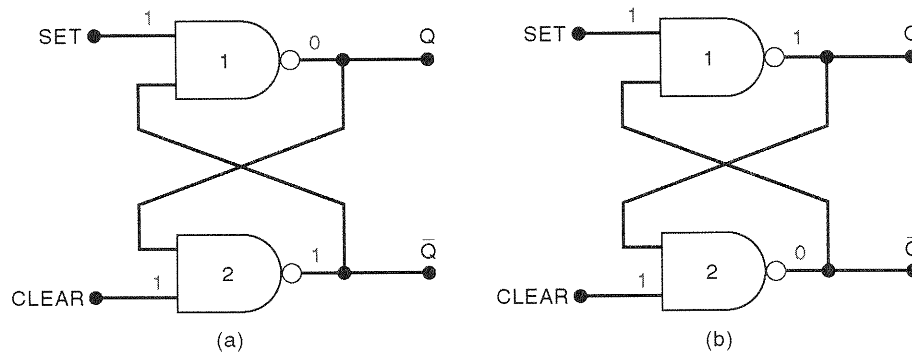


Fig. 5-3 Um latch NAND tem dois estados de repouso possíveis quando SET = CLEAR = 1.

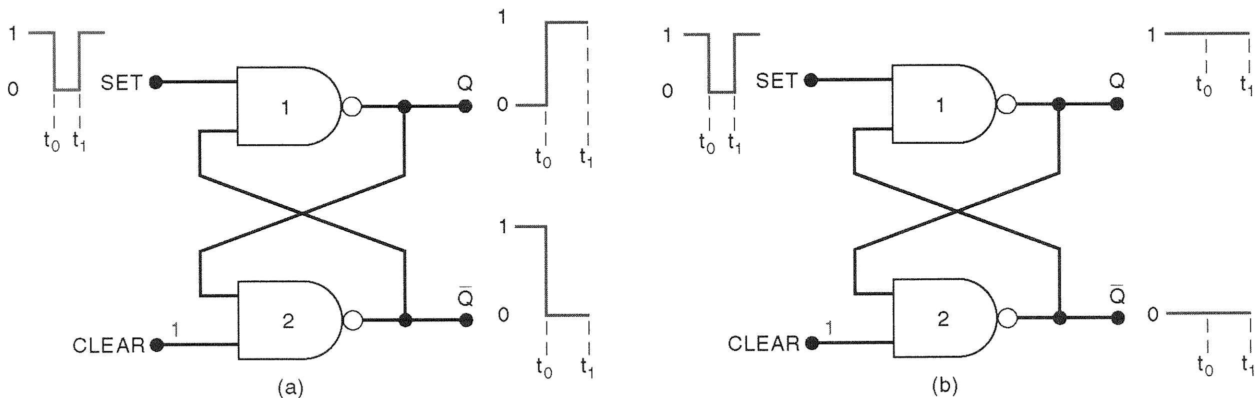


Fig. 5-4 Pulsando-se a entrada SET para o estado 0 quando (a) $Q = 0$ antes do pulso em SET; (b) $Q = 1$ antes do pulso em SET. Note que em ambos os casos Q termina em ALTO.

A Fig. 5-5(b) ilustra a situação onde $Q = 1$ antes da ocorrência do pulso no CLEAR. Assim que o CLEAR vai para BAIXO em t_0, $\overline{Q}$ vai para ALTO, e este nível ALTO força Q a ir para BAIXO, de modo que NAND-2 agora tem as duas entradas em BAIXO. Portanto, quando o CLEAR retorna para ALTO em t_1, a saída da NAND-2 *permanece* em ALTO, o que por sua vez mantém a saída da NAND-1 em BAIXO.

A Fig. 5-5 pode ser resumida considerando-se que um pulso BAIXO na entrada CLEAR sempre leva o latch para o estado onde $Q = 0$. Esta operação é denominada *limpar* ou *ressetar* o latch.

SET e CLEAR Ativos Simultaneamente

O último caso a ser considerado é aquele em que as entradas SET e CLEAR são simultaneamente pulsadas em BAIXO. Isto produz um nível ALTO nas saídas das duas portas NAND, de modo que $Q = \overline{Q} = 1$. É claro que isto é uma

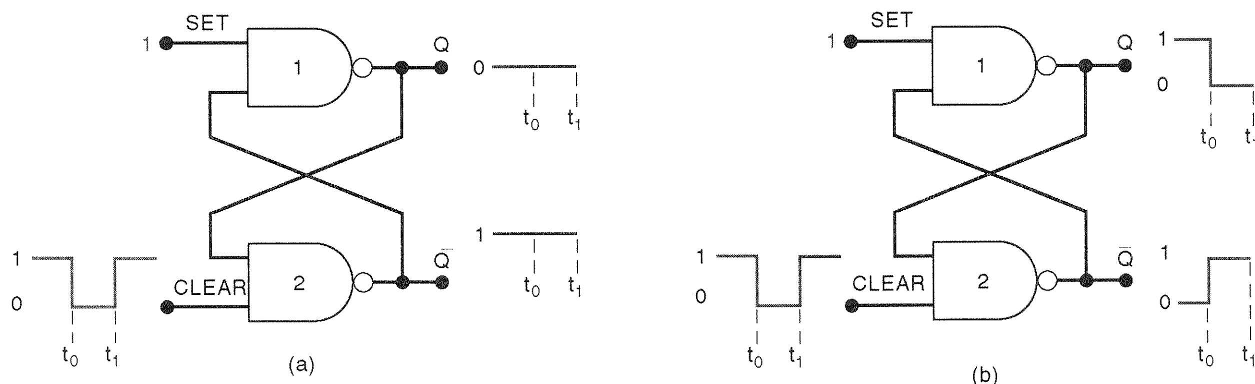


Fig. 5-5 Pulsando-se a entrada CLEAR para o estado BAIXO quando (a) $Q = 0$ antes do pulso em CLEAR; (b) $Q = 1$ antes do pulso em CLEAR. Em ambos os casos, Q termina em BAIXO.

condição indesejada, visto que as duas saídas supostamente são complementares entre si. Além disso, quando as entradas SET e CLEAR retornam para ALTO, o estado de saída resultante dependerá da entrada que voltar a ALTO primeiro. Transições simultâneas de volta para 1 produzirão resultados imprevisíveis. Por estas razões, a condição SET = CLEAR = 0 normalmente não é utilizada para o latch NAND.

Resumo do Latch NAND

A operação descrita anteriormente pode ser convenientemente colocada em uma tabela-verdade (Fig. 5-6) e é resumida a seguir:

1. **SET = CLEAR = 1.** Esta condição é o estado normal de repouso e não tem nenhum efeito sobre o estado de saída. As saídas Q e $\overline{Q}$ permanecerão com os mesmos valores que estavam antes desta condição de entrada.
2. **SET = 0, CLEAR = 1.** Isto sempre faz a saída ir para o estado no qual $Q = 1$, onde permanecerá mesmo após SET retornar para ALTO. Isto é denominado *setar* o latch.
3. **SET = 1, CLEAR = 0.** Isto sempre produz o estado $Q = 0$, onde a saída permanecerá mesmo após CLEAR retornar para ALTO. Isto é denominado *limpar* ou *ressetar* o latch.
4. **SET = CLEAR = 0.** Esta condição tenta setar e limpar o latch ao mesmo tempo e pode produzir resultados ambíguos. Não deve ser usada.

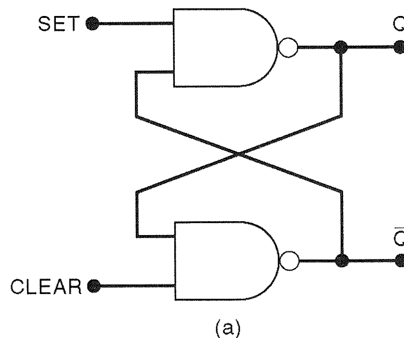
Representações Alternativas

Considerando-se a descrição da operação do latch NAND, deve ficar claro que as entradas de SET e CLEAR são ativas em BAIXO. A entrada SET faz $Q = 1$ quando SET vai para BAIXO, e a entrada CLEAR faz $Q = 0$ quando CLEAR vai para BAIXO. Por causa disto, o latch NAND freqüentemente é desenhado usando-se a representação alternativa para cada porta NAND, conforme mostra a Fig. 5-7(a). As bolhas nas entradas, assim como a identificação dos sinais como $\overline{SET}$ e $\overline{CLEAR}$, indicam o estado de acionamento BAIXO para estas entradas.

A Fig. 5-7(b) mostra uma representação simplificada que usaremos algumas vezes. As letras *S* e *C* representam as entradas SET e CLEAR, enquanto as bolhas indicam que essas entradas são ativas em nível BAIXO. Sempre que este símbolo for usado, ele representa um latch NAND.

Terminologia

A ação de *limpar* um FF ou latch também é chamada *ressetar*, e ambos os termos são usados indistintamente na área digital. Na verdade, a entrada CLEAR também pode ser chamada de RESET, e um latch SET-CLEAR pode ser denominado latch SET-RESET.



Set	Clear	Saída
1	1	Não muda
0	1	Q = 1
1	0	Q = 0
0	0	Inválido*

* produz $Q = \overline{Q} = 1$

(b)

Fig. 5-6 (a) Latch NAND; (b) tabela-verdade.

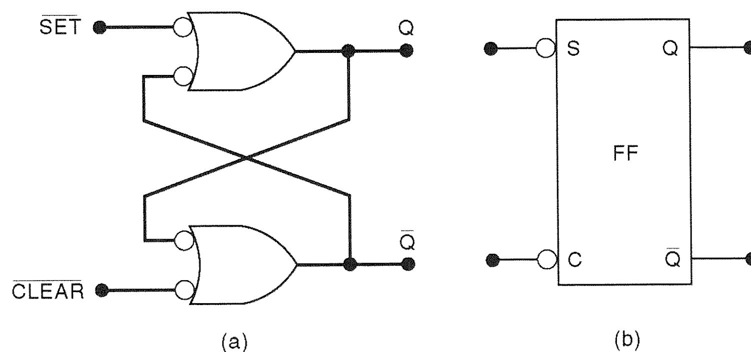


Fig. 5-7 (a) Representação equivalente para o latch NAND; (b) símbolo simplificado.

EXEMPLO 5-1

As formas de onda da Fig. 5-8 são aplicadas nas entradas do latch da Fig. 5-7. Considere que inicialmente $Q = 0$ e determine a forma de onda de Q.

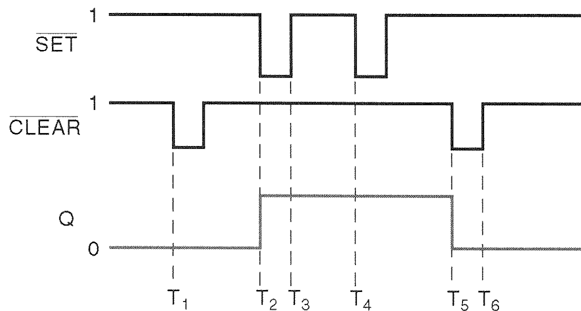


Fig. 5-8 Exemplo 5-1.

Solução

Inicialmente, $\overline{SET} = \overline{CLEAR} = 1$ e portanto Q permanecerá no estado 0. O pulso em BAIXO que ocorre na entrada CLEAR em T_1 não tem efeito algum, pois Q já está no estado 0.

O único modo de Q ir para o estado 1 é através de um pulso em BAIXO na entrada $\overline{SET}$. Isto acontece no instante T_2, quando $\overline{SET}$ vai para BAIXO pela primeira vez. Quando $\overline{SET}$ retorna para ALTO em T_3, Q permanece no seu novo estado ALTO.

No instante de tempo T_4, quando $\overline{SET}$ vai para BAIXO outra vez, não há efeito sobre Q pois Q já está em 1.

O único modo de trazer Q de volta para o estado 0 é através de um pulso em BAIXO na entrada $\overline{CLEAR}$. Isto ocorre em T_5. Quando $\overline{CLEAR}$ retorna para 1 no instante de tempo T_6, Q permanece no estado BAIXO.

Este exemplo mostra que a saída do latch "lembra" a última entrada que foi ativada, e que não muda de estado até que a entrada oposta seja acionada.

EXEMPLO 5-2

É praticamente impossível obter uma transição de tensão "limpa" com uma chave mecânica, por causa do fenômeno conhecido como **trepidação de contato** (**contact bounce**). Isto é ilustrado na Fig. 5-9(a), onde a ação de mover a chave da posição de contato 1 para a 2 produz muitas transições na tensão de saída, conforme a chave trepida (faz e interrompe o contato com 2 muitas vezes) antes de parar sobre o contato 2.

As múltiplas transições no sinal de saída geralmente não duram mais do que uns poucos milissegundos, mas podem ser inaceitáveis em muitas aplicações. Um latch NAND pode ser usado para evitar que a presença da trepidação de contato afete a saída. Descreva a operação do circuito da Fig. 5-9(b) utilizado para eliminar os efeitos da trepidação de contato.

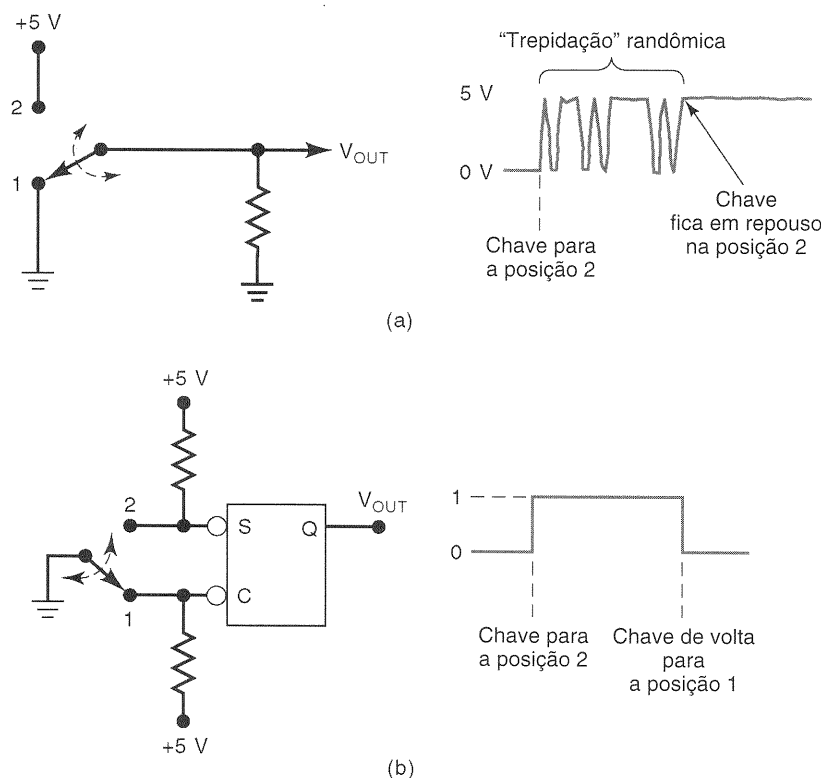


Fig. 5-9 (a) Trepidação mecânica do contato produz múltiplas transições; (b) latch NAND usado para eliminar as múltiplas transições.

Solução

Suponha que a chave está em repouso na posição 1, de modo que a entrada $\overline{\text{CLEAR}}$ está em BAIXO e $Q = 0$. Quando a chave é levada para a posição 2, $\overline{\text{CLEAR}}$ vai para ALTO, e um nível BAIXO aparece na entrada $\overline{\text{SET}}$ quando a chave faz o primeiro contato. Isto faz $Q = 1$ dentro de poucos nanossegundos (o tempo de resposta da porta NAND). Agora, caso a chave desfaça o contato com 2, $\overline{\text{SET}}$ e $\overline{\text{CLEAR}}$ estarão em ALTO, e Q não é afetada, permanecendo em ALTO. Portanto, nada acontece com Q conforme a chave trepida no contato 2 antes de ficar finalmente em repouso na posição 2.

Analogamente, quando a chave é levada de volta da posição 2 para a posição 1, ela coloca um nível BAIXO na entrada $\overline{\text{CLEAR}}$ logo ao primeiro contato. Isto limpa Q para o estado BAIXO, onde permanece mesmo que a chave trepide no contato 1 muitas vezes antes de ficar em repouso.

Deste modo, a saída Q apresenta uma única transição cada vez que a chave é levada de uma posição para a outra.

Questões de Revisão

1. Qual é o estado normal de repouso das entradas $\overline{\text{SET}}$ e $\overline{\text{CLEAR}}$? Qual é o estado ativo de cada entrada?
2. Qual é o estado de Q e $\overline{Q}$ após um FF ter sido limpo (ressetado)?
3. *Verdadeiro ou falso:* A entrada $\overline{\text{SET}}$ nunca pode ser usada para fazer $Q = 0$.
4. Quando a alimentação é inicialmente aplicada em qualquer circuito com FFs, é impossível prever os estados iniciais de Q e $\overline{Q}$. O que poderia ser feito para garantir que um latch NAND sempre comece no estado $Q = 1$?

5-2 LATCH COM PORTAS NOR

Duas portas NOR interligadas de modo cruzado podem ser usadas como um **latch com portas NOR**. A configuração mostrada na Fig. 5-10(a) é similar ao latch NAND, exceto que as saídas Q e $\overline{Q}$ estão em posições trocadas.

A análise da operação do latch NOR pode ser feita exatamente do mesmo modo que aquela feita para o latch NAND. Os resultados estão apresentados na tabela-verdade da Fig. 5-10(b) e podem ser resumidos como segue:

1. **SET = CLEAR = 0.** Este é o estado de repouso normal para o latch NOR, e não provoca nenhum efeito no estado de saída. Q e $\overline{Q}$ permanecem com os mesmos valores que estavam antes da ocorrência desta condição de entrada.
2. **SET = 1, CLEAR = 0.** Isto sempre leva a $Q = 1$, onde permanece mesmo após SET retornar a 0.
3. **SET = 0, CLEAR = 1.** Isto sempre limpa $Q = 0$, onde permanece mesmo após CLEAR retornar a 0.
4. **SET = CLEAR = 1.** Esta condição tenta setar e limpar o latch ao mesmo tempo, e produz $Q = \overline{Q} = 0$. Se as entradas retornam a 0 simultaneamente, o estado de saída resultante é imprevisível. Esta condição de entrada não deve ser usada.

O latch com portas NOR opera exatamente como um latch com portas NAND, exceto que as entradas SET e CLEAR são ativas em ALTO em vez de ativas em BAIXO e que o estado normal de repouso é SET = CLEAR = 0. Q é levado ao nível ALTO por um pulso ALTO na entrada SET e é colocado em BAIXO por um pulso ALTO na entrada CLEAR. O símbolo simplificado para o latch NOR na Fig. 5-10(c) é apresentado sem as bolhas nas entradas S e C, e isto indica que estas entradas são ativas em ALTO.

EXEMPLO 5-3

Suponha que inicialmente $Q = 0$, e determine a forma de onda de Q para as entradas do latch NOR da Fig. 5-11.

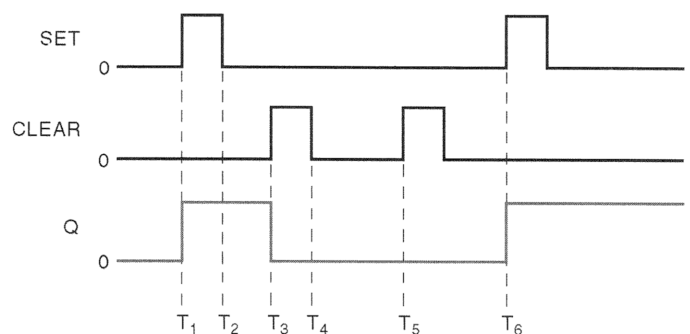
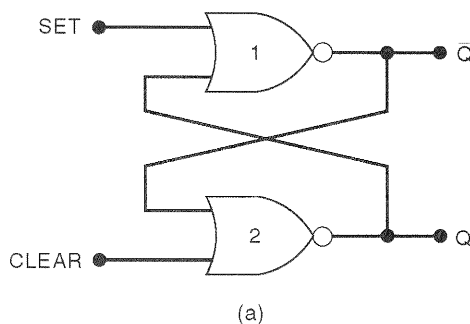


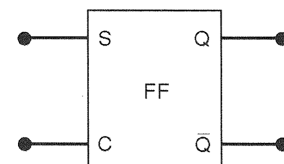
Fig. 5-11 Exemplo 5-3.



Set	Clear	Saída
0	0	Não muda
1	0	Q = 1
0	1	Q = 0
1	1	Inválido*

* produz $Q = \overline{Q} = 0$

(b)



(c)

Fig. 5-10 (a) Latch com portas NOR; (b) tabela-verdade; (c) símbolo simplificado.

Solução

Inicialmente SET = CLEAR = 0, que não afeta a saída, e Q fica em BAIXO. Quando SET vai para ALTO no instante T_1, Q vai para 1 e permanece lá mesmo depois de SET voltar a 0 em T_2.

Em T_3 a entrada CLEAR vai para ALTO e limpa Q para o estado 0, onde permanece mesmo após CLEAR retornar para BAIXO em T_4.

O pulso de CLEAR em T_5 não tem efeito em Q, pois esta saída já está em nível BAIXO. O pulso de SET em T_6 novamente seta Q de volta a 1, onde permanece.

Este exemplo mostra que o FF "lembra" a última entrada que foi ativada, e não muda de estado até que a entrada oposta seja ativada.

EXEMPLO 5-4

A Fig. 5-12 mostra um circuito simples que pode ser usado para detectar a interrupção de um feixe de luz. A luz é focalizada em um fototransistor, que está conectado na configuração emissor comum para operar como uma chave. Suponha que o latch foi previamente limpo para o estado 0 abrindo-se a chave SW1 momentaneamente, e descreva o que acontece se o feixe de luz for momentaneamente interrompido.

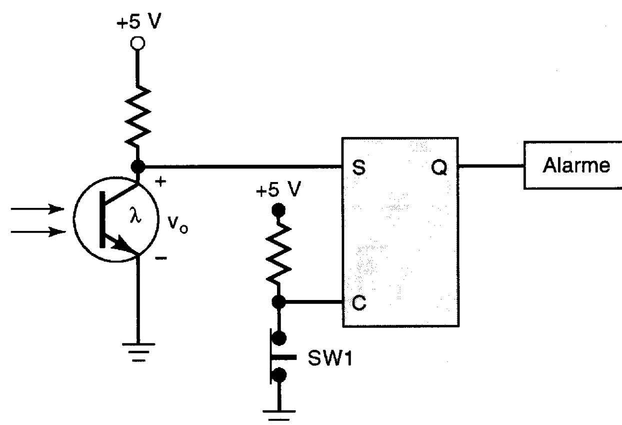


Fig. 5-12 Exemplo 5-4.

Solução

Com a luz incidindo no fototransistor, podemos supor que ele conduz totalmente (satura), de modo que a resistência entre coletor e emissor é muito pequena. Logo, v_0 fica próximo de 0 V. Isto representa um nível BAIXO na entrada SET do latch, de modo que SET = CLEAR = 0.

Quando o feixe de luz é interrompido, o fototransistor corta, e sua resistência coletor-emissor se torna muito alta (isto é, essencialmente um circuito aberto). Isto faz com que v_0 alcance aproximadamente 5 V e ativa a entrada SET que seta Q em ALTO e liga o alarme.

A saída Q permanece em ALTO, e o alarme continua ligado, mesmo que v_0 retorne a 0 V (isto é, mesmo que o feixe de luz seja interrompido apenas momentaneamente), pois SET e CLEAR estariam ambos em BAIXO, o que não provocaria mudança em Q.

Nesta aplicação, a propriedade de memória do latch é usada para converter uma ocorrência momentânea (a interrupção do feixe) em uma saída constante.

Estado do Flip-Flop quando a Alimentação É Ligada

Quando ligamos a fonte de alimentação de um circuito, não é possível prever o estado inicial de uma saída de um flip-flop se as entradas SET e CLEAR estiverem em seus estados inativos (por exemplo, $S = C = 1$ para um latch NAND, $S = C = 0$ para um latch NOR). Existem chances iguais de o estado inicial ser $Q = 0$ ou $Q = 1$. O estado inicial dependerá de fatores tais como: atrasos de propagação internos, capacitâncias parasitas e carregamento externo. Se um latch, ou um FF, deve começar em um determinado estado para garantir a correta operação de um circuito, então ele deve ser colocado neste estado ativando-se momentaneamente a entrada SET ou CLEAR no início da operação do circuito. Freqüentemente isto é conseguido pela aplicação de um pulso na entrada apropriada.

Questões de Revisão

1. Qual é o estado normal de repouso das entradas de um latch NOR? Qual é o estado ativo?
2. Quando um FF é setado, quais são os estados de Q e $\overline{Q}$?
3. Qual é o único modo de fazer com que a saída Q de um latch NOR mude de 1 para 0?
4. Se o latch NOR da Fig. 5-12 fosse substituído por um latch NAND, por que o circuito não funcionaria corretamente?

5-3 ESTUDO DE CASOS EM PESQUISA DE FALHAS

Os dois exemplos a seguir mostrarão o tipo de raciocínio que é empregado na depuração de um circuito que contém um latch.

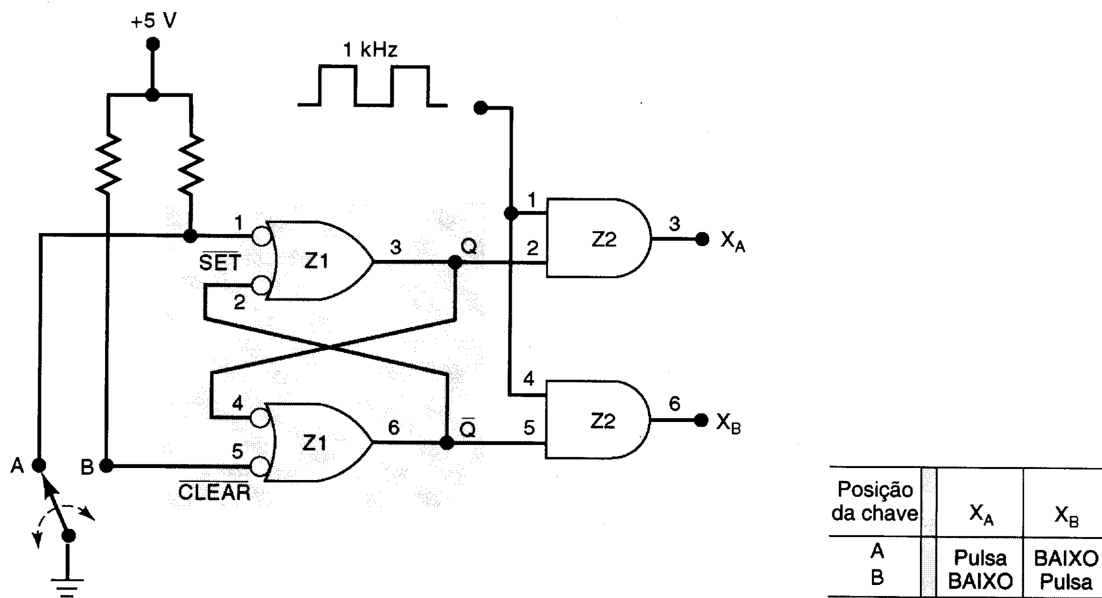
EXEMPLO 5-5

Analise e descreva a operação do circuito da Fig. 5-13.

Solução

A chave é usada para setar ou ressetar o latch NAND produzindo sinais livres de trepidação nas saídas Q e $\overline{Q}$. As saídas deste latch controlam a passagem de uma onda quadrada de 1kHz para as saídas X_A e X_B das portas AND.

Quando a chave se move para a posição A, o latch vai para o estado onde $Q = 1$. Isto permite que a onda quadra-



Posição da chave	X_A	X_B
A	Pulsa	BAIXO
B	BAIXO	Pulsa

Fig. 5-13 Exemplos 5-5 e 5-6.

da de 1 kHz chegue a X_A, enquanto o nível BAIXO em $\overline{Q}$ mantém $X_B = 0$. Quando a chave vai para a posição *B*, o latch é limpo ($Q = 0$), o que mantém X_A em 0, enquanto o nível ALTO em $\overline{Q}$ permite que a onda chegue a X_B.

EXEMPLO 5-6

Um estudante testa o circuito da Fig. 5-13 e anota suas observações sobre o estado de vários pontos do circuito, conforme está mostrado na Tabela 5-1. Ele observa que, quando a chave está na posição *B*, o circuito funciona corretamente; entretanto, quando ela está na posição *A*, o latch não vai para o estado onde $Q = 1$. Quais são as possíveis causas deste mau funcionamento?

Solução

Existem várias possibilidades, como podemos ver a seguir:

1. Um circuito aberto interno em Z1-1. Isto impediria *Q* de responder à entrada $\overline{\text{SET}}$.
2. Um componente interno danificado na porta NAND Z1 que o impediria de responder de modo correto.
3. A saída *Q* está permanentemente em nível BAIXO. Isto pode ser causado por:
   (a) Z1-3 internamente em curto com a terra
   (b) Z1-4 internamente em curto com a terra
   (c) Z2-2 internamente em curto com a terra
   (d) O ponto *Q* externamente em curto com a terra

Uma verificação com o ohmímetro vai determinar se alguma destas condições está presente, e uma inspeção visual deve revelar a existência de curto externo.

O que dizer sobre a hipótese de $\overline{Q}$ estar internamente ou externamente em curto com V_{CC}? Um pouco de raciocínio vai levá-lo à conclusão de que esta falha não pode ser a causa do problema. Se $\overline{Q}$ estivesse em curto com V_{CC}, não haveria impedimento para que a saída *Q* fosse para ALTO quando $\overline{\text{SET}}$ fosse para BAIXO. Uma vez que a saída *Q não vai* para ALTO, esta não pode ser a causa do problema. $\overline{Q}$ parece estar permanentemente em ALTO pois *Q* está permanentemente em BAIXO, o que mantém $\overline{Q}$ em ALTO.

5-4 SINAIS DE CLOCK E FLIP-FLOPS COM CLOCK

Sistemas digitais podem operar de modo assíncrono ou de modo síncrono. Nos sistemas assíncronos, as saídas dos circuitos lógicos podem mudar de estado a qualquer momento em que uma ou mais entradas mudem de estado. Um sistema assíncrono é geralmente mais difícil de se projetar e depurar do que um sistema síncrono.

Nos sistemas síncronos, um sinal, comumente chamado **clock** (relógio), determina os momentos nos quais qualquer

TABELA 5-1

Posição da chave	$\overline{\text{SET}}$ (Z1-1)	$\overline{\text{CLEAR}}$ (Z1-5)	Q (Z1-3)	$\overline{Q}$ (Z1-6)	X_A (Z2-3)	X_B (Z2-6)
A	BAIXO	ALTO	BAIXO	ALTO	BAIXO	Pulsa
B	ALTO	BAIXO	BAIXO	ALTO	BAIXO	Pulsa

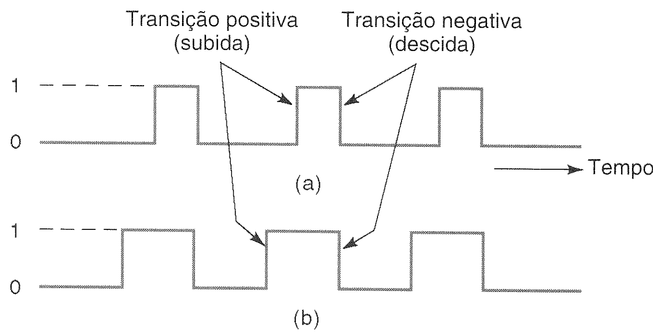


Fig. 5-14 Sinais de clock.

uma das saídas pode mudar de estado. Este sinal de clock é geralmente um trem de pulsos retangulares, ou uma onda quadrada, como pode ser visto na Fig. 5-14. O sinal de clock é distribuído para todas as partes do sistema, e a maioria das saídas (senão todas) do sistema pode mudar de estado somente quando o clock faz uma transição. As transições (também chamadas de bordas) estão indicadas na Fig. 5-14. Quando o clock faz uma transição de 0 para 1, esta é chamada de **transição positiva (subida).** Quando o clock faz uma transição de 1 para 0, esta é chamada de **transição negativa (descida).**

A maioria dos sistemas é síncrona, embora existam sempre algumas partes assíncronas, porque circuitos síncronos são mais fáceis de projetar e depurar. Eles são mais fáceis de depurar porque as saídas dos circuitos podem mudar de estado apenas em instantes de tempo bem determinados. Em outras palavras, quase tudo está sincronizado com as transições do sinal de clock.

A sincronização feita pelos sinais de clock é obtida através do uso de **flip-flops com clock** que são projetados para mudar de estado em uma das transições do clock.

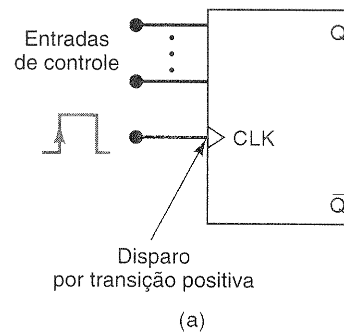
Flip-Flops com Clock

Vários tipos de FFs com clock são usados em um grande número de aplicações. Antes de iniciarmos nosso estudo dos diferentes tipos de FFs, descreveremos os conceitos fundamentais que são comuns a todos eles.

1. Flip-flops com clock têm uma entrada de clock que é geralmente chamada de *CLK*, *CK* ou *CP*. Normalmente usaremos *CLK*, como mostrado na Fig. 5-15. Na maioria dos FFs com clock, a entrada *CLK* é **disparada por transição,*** o que significa que ela é ativada pela transição do sinal presente nesta entrada. Isto é indicado por um pequeno triângulo na entrada *CLK*. Isto diferencia os flip-flops dos latches que são disparados por nível.

Na Fig. 5-15(a), a entrada *CLK* é ativada apenas quando uma transição positiva ocorre, não sendo ativada em nenhum outro momento. Na Fig. 5-15(b), a entrada *CLK* é ativada apenas por uma transição negativa, que é simbolizada pela presença de uma pequena bolha.

*É bastante comum o uso dos termos "gatilhada por transição", "gatilhada por borda", "sensível a borda", "trigada pela borda" e "trigada por transição" como sinônimos de disparada por transição. A palavra trigada é oriunda da palavra inglesa *triggered*, que significa disparado. (*N. do T.*)



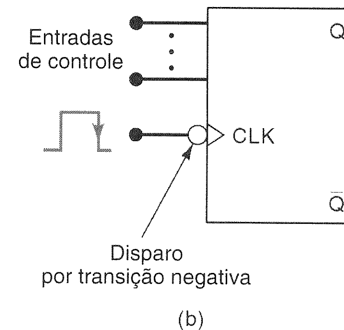


Fig. 5-15 Flip-flops com clock têm uma entrada de clock que pode ser disparada por (a) uma transição positiva ou (b) uma transição negativa. As entradas de controle determinam o efeito da transição de disparo.

2. FFs com clock também possuem uma ou mais **entradas de controle** que podem ter vários nomes, dependendo do seu funcionamento. As entradas de controle não têm efeito algum sobre *Q* até que ocorra uma transição de disparo na entrada *CLK*. Em outras palavras, seu efeito sobre *Q* é sincronizado com o sinal aplicado a *CLK*. Por esta razão, elas são chamadas de **entradas de controle síncronas**.

Por exemplo, as entradas de controle do FF vistas na Fig. 5-15(a) não afetam *Q* até que uma transição positiva do sinal de clock ocorra. Do mesmo modo, as entradas de controle da Fig. 5-15(b) não afetarão *Q* enquanto não ocorrer uma transição negativa do sinal de clock.

3. Resumindo, podemos dizer que as entradas de controle deixam as saídas dos flip-flops prontas para mudar de estado, enquanto a transição ativa na entrada *CLK*, de fato, *dispara* esta mudança. As entradas de controle são responsáveis pelo QUE deve mudar (isto é, para que estado a saída deve ir), enquanto a entrada *CLK* determina QUANDO isto deve acontecer.

Tempos de Setup (Preparação) e Hold (Manutenção)

Dois parâmetros de temporização devem ser observados para que o FF responda de modo confiável às suas entradas de controle quando ocorrer uma transição de disparo na entrada *CLK*. Estes parâmetros são ilustrados na Fig. 5-16 para um FF disparado por transição positiva.

O **tempo de setup, t_S,** é o intervalo de tempo que precede imediatamente uma transição ativa do sinal de *CLK*,

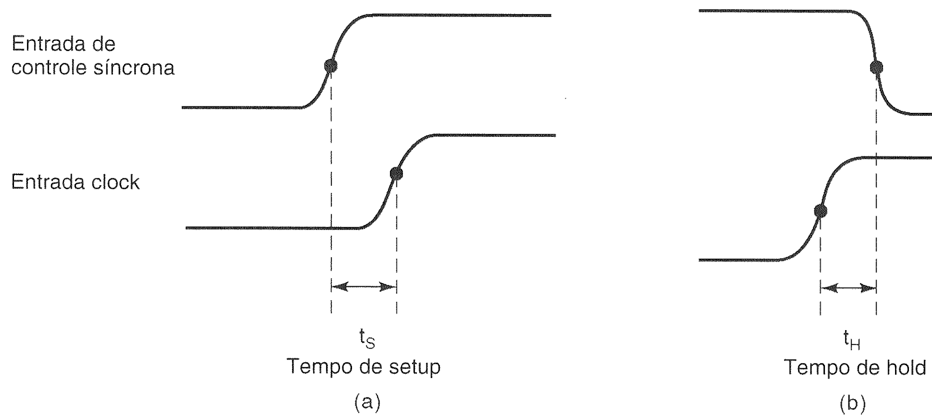


Fig. 5-16 Entradas de controle devem ser mantidas estáveis por (a) um tempo t_S antes da transição de disparo e por (b) um tempo t_H após a transição de disparo.

durante o qual cada entrada de controle deve permanecer em um nível estável. Os fabricantes de CIs geralmente especificam o tempo mínimo de setup permitido t_S(min). Se este parâmetro não for respeitado, o FF pode não responder de modo confiável quando houver uma transição do clock.

O **tempo de hold, t_H,** é o intervalo de tempo que se segue imediatamente após uma transição de disparo do sinal de *CLK*, durante o qual as entradas de controle síncronas devem ser mantidas em um nível estável. Os fabricantes de CIs geralmente especificam um valor mínimo aceitável para o tempo de hold, t_H(min). Se este parâmetro não for respeitado, o FF pode não responder de modo confiável quando houver uma transição do clock.

Assim, para garantir que um FF com clock responda de modo correto quando ocorrer uma transição de disparo do clock, as entradas de controle devem estar estáveis, isto é, não devem mudar de estado, pelo menos durante um intervalo de tempo igual a t_S(min) *antes* da transição do clock, e pelo menos por um intervalo igual a t_H(min) depois da transição do clock.

CIs de flip-flops têm valores mínimos permitidos para t_S e t_H da ordem de nanossegundos. Os tempos de setup são geralmente da ordem de 5 a 50 ns, enquanto os tempos de hold são geralmente da ordem de 0 a 10 ns. Observe que estes intervalos são medidos nos instantes em que as transições estão em 50%.

Esses parâmetros são muito importantes em sistemas síncronos porque, como veremos, existirão muitas situações em que as entradas de controle síncronas estarão mudando de estado aproximadamente ao mesmo tempo que a entrada *CLK*.

Questões de Revisão

1. Quais são os dois tipos de entrada que um FF possui?
2. O que significa o termo *disparado por transição*?
3. *Verdadeiro ou falso*: A entrada *CLK* afetará a saída do FF apenas quando a transição ativa das entradas de controle ocorrer.
4. Defina os parâmetros de tempo de setup e tempo de hold de um FF com clock.

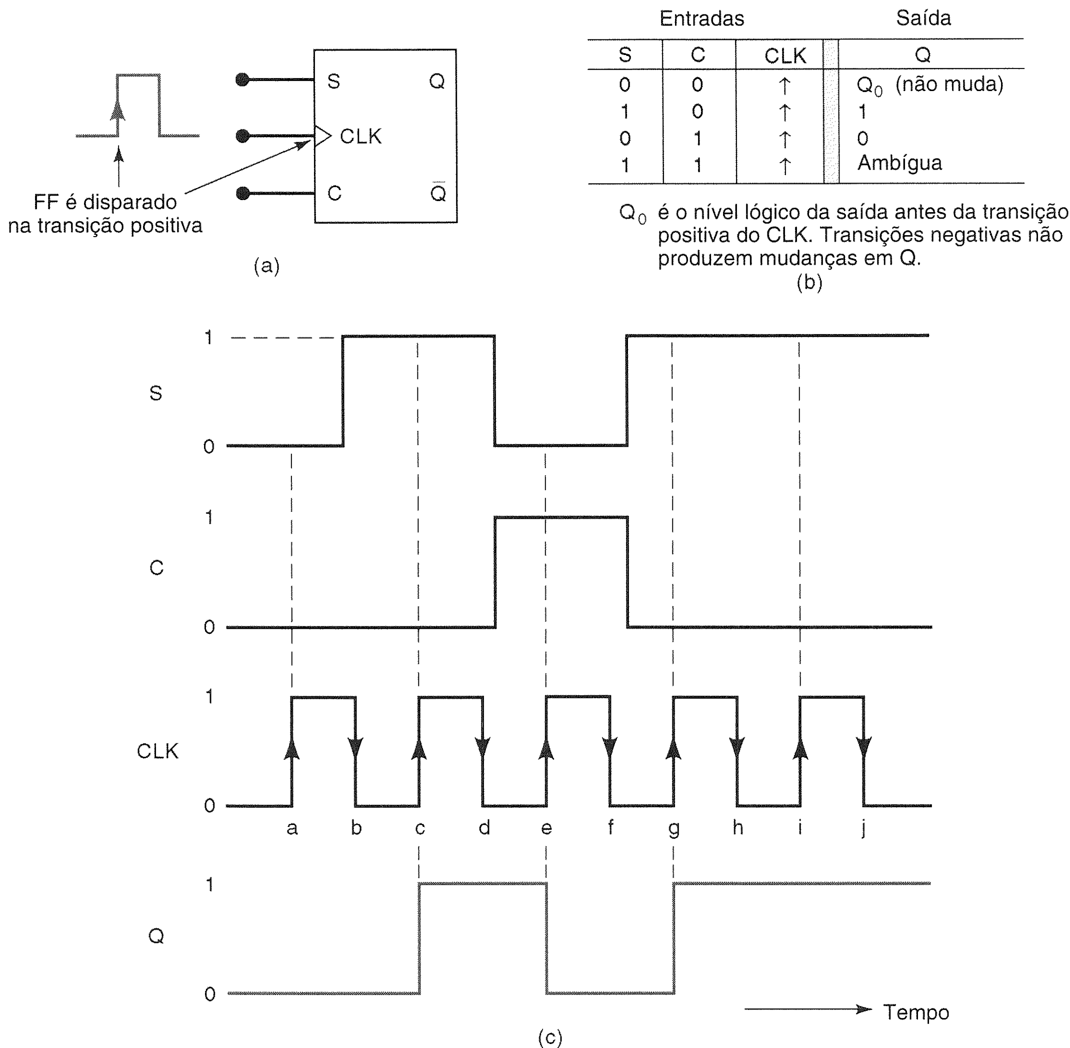
5-5 FLIP-FLOP S-C COM CLOCK

A Fig. 5-17(a) mostra o símbolo lógico de um **flip-flop S-C com clock** que é disparado pela transição positiva do sinal de clock. Isto significa que o flip-flop pode mudar de estado *somente* quando o sinal aplicado na sua entrada *CLK* faz uma transição de 0 para 1. As entradas *S* e *C* controlam o estado do FF, do mesmo modo que foi descrito anteriormente para o caso do latch com portas NOR; entretanto, a saída do flip-flop não responderá a estas entradas até a ocorrência de uma transição positiva do sinal de clock.

A tabela-verdade na Fig. 5-17(b) mostra como a saída do flip-flop responde a uma transição positiva na entrada *CLK* para várias combinações possíveis das entradas *S* e *C*. Esta tabela-verdade utiliza uma nova nomenclatura. A seta para cima (↑) indica que uma transição positiva é necessária na entrada *CLK*. Q_0 representa o nível lógico existente antes da transição positiva. Esta nomenclatura é amplamente utilizada pelos fabricantes nos manuais de circuitos integrados.

As formas de onda na Fig. 5-17(c) mostram a operação de um flip-flop S-C. Se admitirmos que os tempos de setup e hold são respeitados em todos os casos, podemos analisar as formas de onda como segue:

1. Inicialmente todas as entradas estão em 0 e vamos supor que a saída Q está em 0, ou seja, $Q_0 = 0$.
2. Quando ocorre a primeira transição positiva do sinal de clock (ponto *a*), as entradas *S* e *C* estão ambas em 0, e portanto o estado do flip-flop não é alterado e permanece no estado $Q = 0$ (isto é, $Q = Q_0$).
3. Quando a segunda transição positiva do sinal de clock ocorre (ponto *c*), a entrada *S* está agora em alto, enquanto *C* permanece em baixo, portanto o FF vai para o estado $Q = 1$ na subida do pulso de clock.
4. Quando o terceiro pulso de clock faz a sua transição positiva (ponto *e*), *S* está em 0 e *C* está em 1, o que faz com que o flip-flop vá para o estado 0.
5. No quarto pulso de clock, o flip-flop vai para o estado $Q = 1$ (ponto *g*) porque $S = 1$ e $C = 0$ quando a transição positiva ocorre.
6. O quinto pulso de clock também encontra $S = 1$ e $C = 0$, quando sua transição positiva ocorre. Entretanto, como Q já está em alto, ele permanece neste estado.



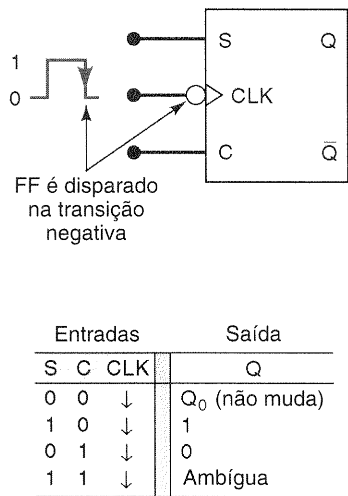
S	C	CLK	Q
0	0	↑	Q_0 (não muda)
1	0	↑	1
0	1	↑	0
1	1	↑	Ambígua

Fig. 5-17 (a) Flip-flop S-C com clock que responde somente às transições positivas dos pulsos de clock; (b) tabela-verdade; (c) formas de onda típicas.

7. A combinação $S = C = 1$ não deve ser usada, pois resulta em uma condição ambígua.

A partir da análise dessas formas de onda, podemos observar que o FF não é afetado pelas transições negativas dos pulsos de clock. Também devemos notar que os níveis lógicos nas entradas *S* e *C* não têm efeito sobre o FF, a não ser que ocorra uma transição positiva do sinal de clock. As entradas *S* e *C* são entradas de *controle* síncronas, pois elas indicam para qual estado o FF deve ir quando ocorrer um pulso de clock. A entrada *CLK* é a entrada de **disparo,*** isto é, a entrada que faz com que o FF mude de estado de acordo com os níveis das entradas *S* e *C*, quando uma transição de disparo no sinal de clock ocorrer.

A Fig. 5-18 mostra o símbolo e a tabela-verdade para um flip-flop S-C com clock que é disparado por uma transição



S	C	CLK	Q
0	0	↓	Q_0 (não muda)
1	0	↓	1
0	1	↓	0
1	1	↓	Ambígua

Fig. 5-18 Flip-flop S-C com clock que é disparado apenas nas transições negativas.

*É comum a utilização do termo original em inglês, *trigger*. (*N. do T.*)

negativa na entrada *CLK*. A pequena bolha junto com o triângulo na entrada *CLK* indica que este FF vai ser disparado quando houver uma transição de 1 para 0 na entrada *CLK*. O FF opera do mesmo modo que aquele disparado por transição positiva, exceto pelo fato de que sua saída muda de estado somente nas transições de descida dos pulsos de clock (pontos *b*, *d*, *f*, *h* e *j* na Fig. 5-17). Tanto os flip-flops disparados por transição negativa quanto aqueles disparados por transição positiva são usados em sistemas digitais.

Circuito Interno de um Flip-Flop S-C Disparado por Transição

Uma análise detalhada do circuito interno de um FF com clock não é necessária, uma vez que todos os tipos estão disponíveis como circuitos integrados. Apesar de nosso interesse estar no funcionamento externo dos flip-flops, podemos compreendê-lo melhor estudando uma versão simplificada dos circuitos internos dos flip-flops. Esta pode ser vista na Fig. 5-19.

O circuito pode ser dividido em três partes principais:

1. Um **latch NAND** formado pelas portas NAND-3 e NAND-4.
2. Um **circuito direcionador de pulsos** formado pelas portas NAND-1 e NAND-2.
3. Um **circuito detector de transição.**

Como pode ser visto na Fig. 5-19, o detector de transição produz um pulso estreito e positivo (*CLK**) que coincide com a transição de disparo na entrada *CLK*. O circuito direcionador de pulsos "direciona" este pulso estreito para a entrada SET ou para a entrada CLEAR, de acordo com os níveis presentes em *S* e *C*. Por exemplo, para $S = 1$ e $C = 0$, o sinal *CLK** passa através da porta NAND-1, sendo invertido e produzindo um pulso em nível BAIXO na entrada SET do latch, o que resulta em $Q = 1$. Com $S = 0$ e $C = 1$, o sinal *CLK** passa através da porta NAND-2, sendo invertido e produzindo um pulso em BAIXO na entrada CLEAR do latch, o que resulta em $Q = 0$.

A Fig. 5-20(a) mostra como o sinal de *CLK** é gerado para flip-flops disparados por transição positiva. O INVERSOR produz um atraso de alguns nanossegundos, de modo que as transições de $\overline{CLK}$ ocorram um pouco depois daquelas de *CLK*. Uma porta AND produz na saída um pulso em ALTO apenas durante os poucos nanossegundos nos quais *CLK* e $\overline{CLK}$ estão ambos em ALTO. Isto resulta em um pulso estrei-

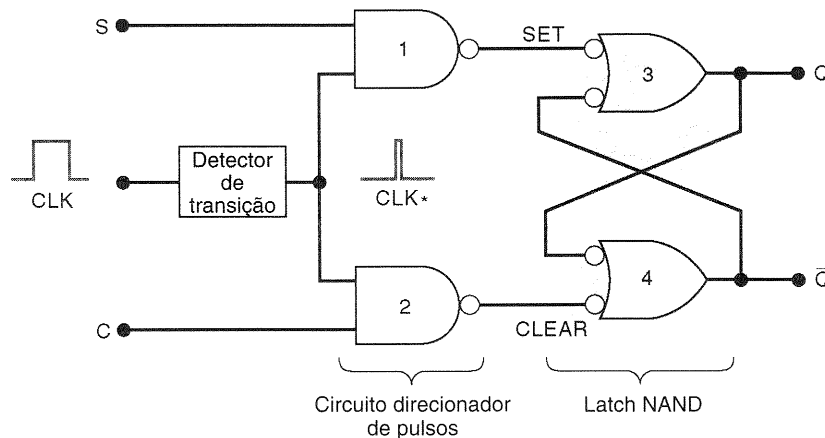


Fig. 5-19 Versão simplificada do circuito interno de um flip-flop disparado por transição.

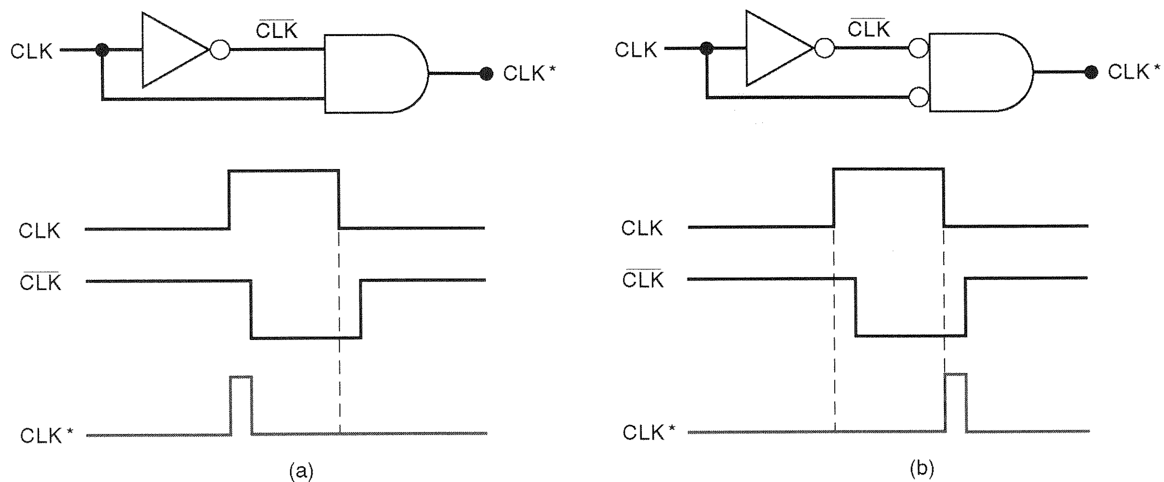


Fig. 5-20 Implementação de circuitos detectores de transição usados em flip-flops disparados por transição: (a) transição negativa; (b) transição positiva. A duração dos pulsos de *CLK** é geralmente de 2 a 5 nanossegundos.

to em *CLK**, que ocorre na subida do sinal de clock. A Fig. 5-20(b) mostra o arranjo necessário para produzir *CLK** na transição negativa para flip-flops que são disparados na descida.

Uma vez que o sinal *CLK** está em ALTO apenas por alguns nanossegundos, a saída *Q* é afetada por *S* e *C* apenas por um curto período, após a ocorrência da transição de disparo de *CLK*. É isto que dá aos flip-flops esta característica de disparo por transição.

Questões de Revisão

1. Suponha que as formas de onda da Fig. 5-17(c) são aplicadas às entradas do flip-flop da Fig. 5-18. O que acontece com *Q* no ponto *b*?
2. Explique por que as entradas *S* e *C* afetam a saída *Q* somente durante a transição de disparo de *CLK*.

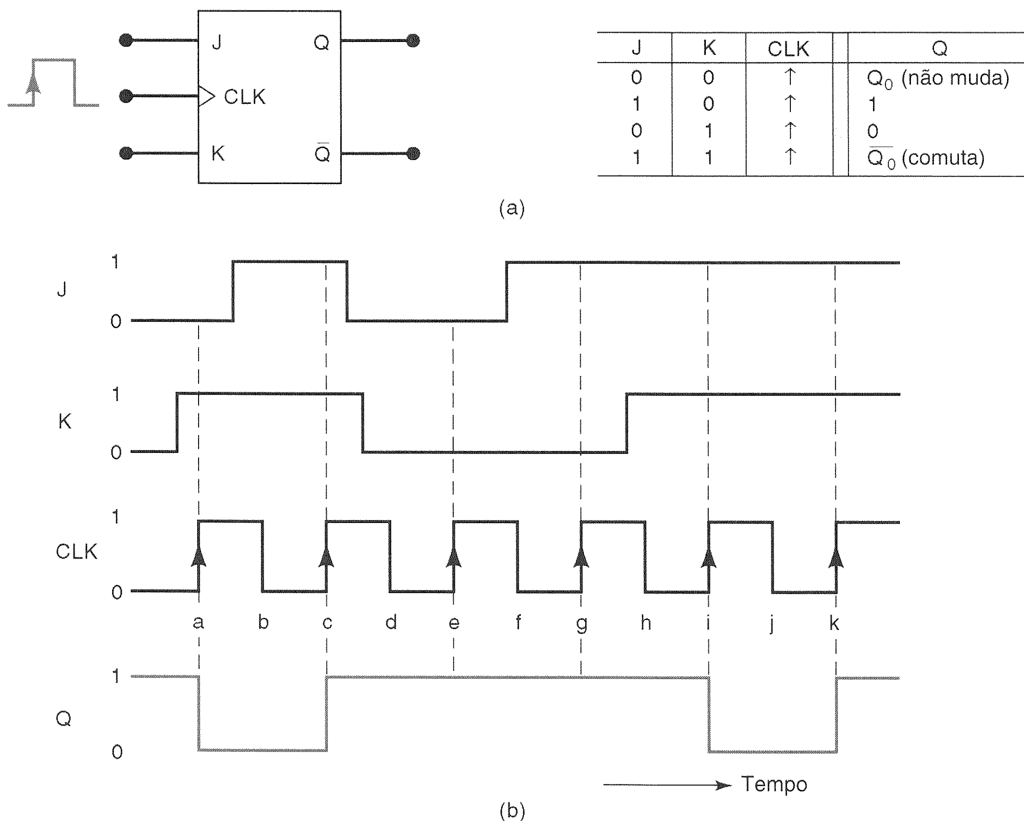
5-6 FLIP-FLOP J-K COM CLOCK

A Fig. 5-21(a) mostra um **flip-flop J-K com clock** que é disparado pela transição positiva do sinal de clock. As entradas *J* e *K* controlam o estado do FF do mesmo modo que as entradas *S* e *C* o fazem para um flip-flop S-C, mas com uma diferença muito importante: *a condição na qual J = K = 1 não resulta em um estado ambíguo na saída.* Para esta condição, o FF sempre irá para o seu estado *oposto* quando a transição positiva ocorrer. Isto é chamado de operação em **modo de comutação** (*toggle mode*). Neste modo, se ambas as entradas *J* e *K* estão em ALTO, o FF muda de estado (comuta) a cada transição positiva do clock.

A tabela-verdade da Fig. 5-21(a) resume como um flip-flop J-K responde à transição positiva para cada combinação de *J* e *K*. Observe que a tabela-verdade é igual à do flip-flop S-C (Fig. 5-17), exceto para a condição $J = K = 1$. Esta condição resulta em $Q = \overline{Q}_0$, o que significa que o novo valor de *Q* será o inverso do que ele tinha antes da transição positiva. Esta operação é chamada de comutação.

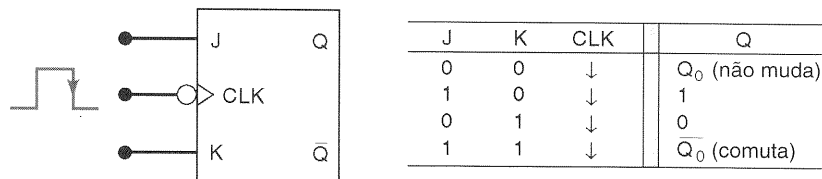
O funcionamento desse flip-flop é ilustrado através das formas de onda da Fig. 5-21(b). Mais uma vez, consideramos que os tempos de setup e hold estão sendo respeitados.

1. Inicialmente, todas as entradas estão em 0, e consideramos que a saída *Q* está em 1; isto é $Q_0 = 1$.
2. Quando a transição positiva do primeiro pulso de clock ocorre (ponto *a*), temos a seguinte condição de entrada: $J = 0$ e $K = 1$ e, portanto, o FF será ressetado (irá para o estado $Q = 0$).
3. O segundo pulso de clock encontra $J = K = 1$ quando faz a sua transição positiva (ponto *c*). Isto faz com que o flip-flop *comute* para seu estado oposto, $Q = 1$.
4. No ponto *e*, tanto *J* como *K* são iguais a 0, e portanto o FF não muda de estado durante esta transição.
5. No ponto *g*, $J = 1$ e $K = 0$. Esta condição faz com que o FF vá para o estado $Q = 1$. Entretanto, *Q* já é igual a 1, e portanto o FF permanece neste estado.
6. No ponto *i*, $J = K = 1$, e portanto o FF comuta para seu estado oposto. A mesma coisa ocorre no ponto *k*.



J	K	CLK	Q
0	0	↑	Q_0 (não muda)
1	0	↑	1
0	1	↑	0
1	1	↑	$\overline{Q}_0$ (comuta)

Fig. 5-21 (a) Flip-flop J-K com clock que responde somente às transições positivas do sinal de clock; (b) formas de onda.



J	K	CLK	Q
0	0	↓	Q_0 (não muda)
1	0	↓	1
0	1	↓	0
1	1	↓	$\overline{Q_0}$ (comuta)

Fig. 5-22 Flip-flop J-K que dispara apenas nas transições negativas.

Analisando essas formas de onda, verificamos que o FF não é afetado pelas transições negativas dos pulsos de clock. Também podemos notar que os níveis presentes nas entradas *J* e *K* somente afetam a saída se uma transição positiva do sinal de clock ocorrer. As entradas *J* e *K* sozinhas jamais poderão alterar o estado do FF.

A Fig. 5-22 mostra o símbolo para um flip-flop com clock que é disparado na descida do sinal de clock. A pequena bolha na entrada *CLK* indica que este FF vai ser disparado quando o sinal na entrada *CLK* fizer uma transição de 1 para 0. Este FF funciona do mesmo modo que o FF disparado por transição positiva mostrado na Fig. 5-21, exceto pelo fato de que a saída vai mudar de estado apenas nas transições negativas do sinal de clock (pontos *b, d, f, h* e *j*). Ambos os tipos de flip-flops são bastante utilizados.

O flip-flop J-K é muito mais versátil que o flip-flop S-C porque não possui estados ambíguos. A condição de entrada onde $J = K = 1$ produz uma operação de comutação, bastante utilizada em todos os tipos de contadores binários. Em resumo, o flip-flop J-K pode fazer tudo que o flip-flop S-C pode, *além* de operar em modo de comutação.

Circuito Interno de um Flip-Flop J-K Disparado por Transição

Uma versão simplificada do circuito interno de um flip-flop J-K disparado por transição pode ser vista na Fig. 5-23. Ela contém as mesmas três partes que possuía o flip-flop S-C disparado por transição (Fig. 5-19). Na verdade, a única diferença entre estes dois circuitos está no fato de que as saídas *Q* e $\overline{Q}$ são realimentadas para o circuito direcionador de pulsos formado pelas portas NAND 1 e 2. Esta realimentação é que fornece ao flip-flop J-K a operação de comutação para a condição onde $J = K = 1$.

Vamos examinar a condição de comutação mais de perto, considerando que J = K = 1 e que a saída *Q* esteja em BAIXO quando o pulso de clock ocorre. Com $Q = 0$ e $\overline{Q} = 1$, a porta NAND 1 vai direcionar *CLK** (invertido) para a entrada $\overline{\text{SET}}$ do latch, fazendo com que a saída *Q* seja igual a 1. Se supusermos que a saída *Q* está em ALTO quando ocorre o pulso de clock, a porta NAND 2 vai direcionar *CLK** (invertido) para a entrada $\overline{\text{CLEAR}}$ do latch para produzir $Q = 0$. Logo, podemos dizer que a saída *Q* sempre vai para o estado oposto.

A fim de que a operação de comutação funcione como foi descrito anteriormente, o pulso *CLK** deve ser muito estreito. Ele deve retornar a 0 antes que as saídas *Q* e $\overline{Q}$ comutem para seus novos valores, pois, caso contrário, os novos valores de *Q* e $\overline{Q}$ vão fazer com que *CLK** comute as saídas do latch mais uma vez.

Questões de Revisão

1. *Falso* ou *verdadeiro*: Um flip-flop J-K pode ser usado como um flip-flop S-C, mas um flip-flop S-C não pode ser usado como um J-K.
2. Um flip-flop J-K tem alguma condição de entrada que seja ambígua?
3. Que condição de entrada para J-K sempre faz $Q = 1$ na ocorrência de uma transição de disparo na entrada *CLK*?

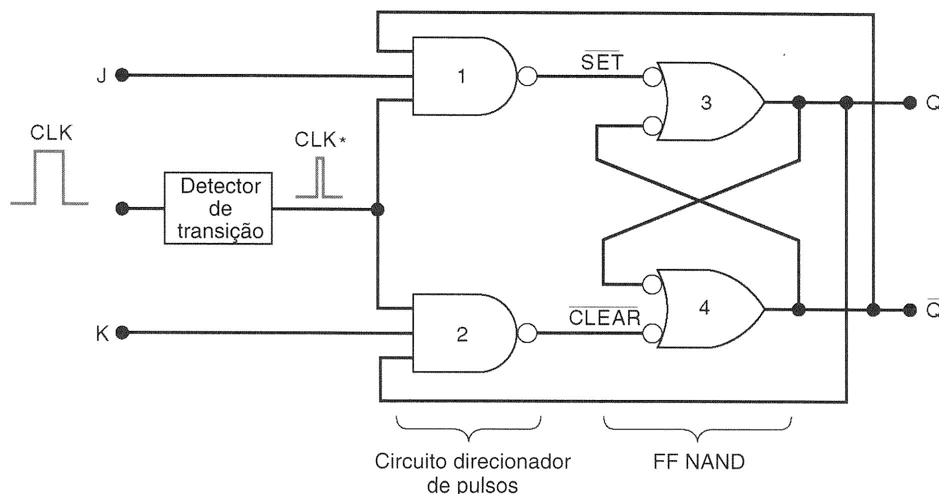


Fig. 5-23 Circuito interno de um flip-flop J-K disparado por transição.

5-7 FLIP-FLOP D COM CLOCK

A Fig. 5-24(a) mostra o símbolo e a tabela-verdade para um **flip-flop D com clock** que é disparado na transição positiva. Ao contrário dos flip-flops J-K e S-C, este flip-flop possui apenas uma entrada de controle síncrona, *D*, que é a inicial da palavra *dados*. A operação do flip-flop *D* é muito simples: *Q* irá para o mesmo estado presente na entrada *D* quando ocorrer uma transição positiva na entrada *CLK*. Em outras palavras, o nível presente em *D* é *armazenado* no flip-flop no instante em que a transição positiva ocorre. As formas de onda na Fig. 5-24(b) mostram esta operação.

Suponha que a saída *Q* está inicialmente em ALTO. Quando a primeira transição positiva ocorre no ponto *a*, a entrada *D* está em BAIXO, logo, *Q* irá para BAIXO. Mesmo que o nível na entrada *D* mude entre os pontos *a* e *b*, a saída não é afetada, pois *Q* está armazenando o nível lógico BAIXO que estava presente na entrada *D* no ponto *a*. Quando a transição positiva ocorre no ponto *b*, *Q* vai para ALTO, uma vez que *D* está em ALTO neste momento. *Q* armazena este nível ALTO até que a transição positiva que ocorre no ponto *c* faz com que a saída *Q* vá para BAIXO, uma vez que *D* está em BAIXO neste momento. De modo semelhante, a saída *Q* assume os níveis presentes na entrada *D* quando ocorre uma transição positiva nos pontos *d*, *e*, *f* e *g*. Observe que a saída *Q* permanece em ALTO, no ponto *e*, porque *D* ainda está em ALTO.

Mais uma vez, é importante lembrar que a saída *Q* pode mudar de estado somente quando ocorrer uma transição positiva. Os valores presentes em *D* no intervalo entre transições positivas não têm influência na saída.

Um flip-flop *D* disparado por transição negativa opera do mesmo modo que acabamos de descrever, exceto pelo fato de que a saída *Q* receberá qualquer valor que estiver em *D* quando ocorrer uma transição negativa na entrada *CLK*. O símbolo para o flip-flop D que dispara em uma transição negativa tem uma pequena bolha na entrada *CLK*.

Implementação de um Flip-Flop D

Um flip-flop D disparado por transição pode ser facilmente implementado adicionando-se um INVERSOR ao flip-flop S-C como está mostrado na Fig. 5-25. Se você aplicar os dois valores possíveis de *D* a este circuito, irá verificar que a saída

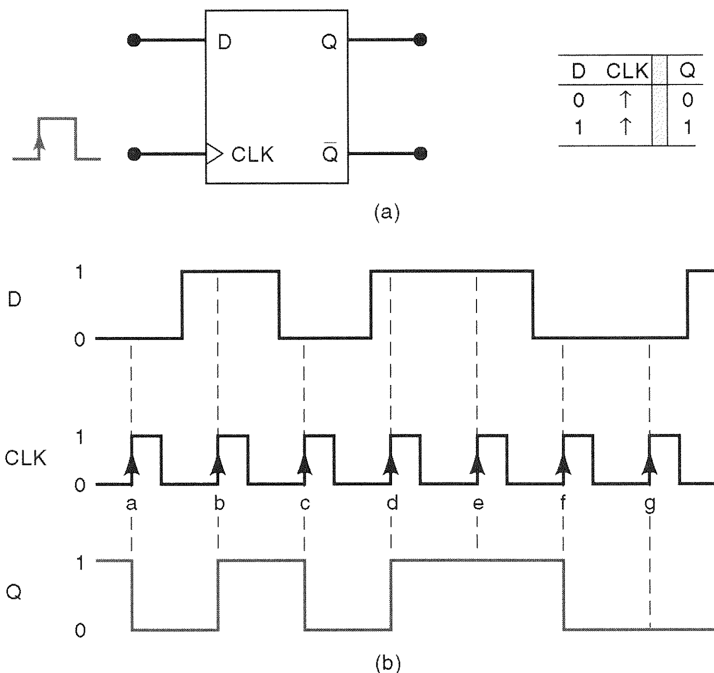


Fig. 5-24 (a) Flip-flop D que dispara apenas nas transições positivas; (b) formas de onda.

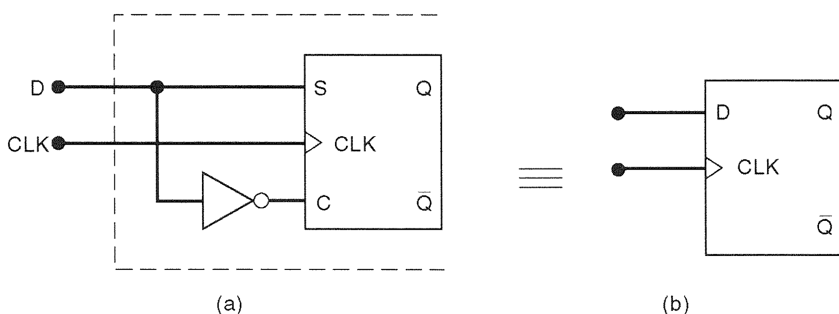


Fig. 5-25 Implementação de um flip-flop D disparado por transição a partir de um flip-flop S-C.

Q assume o nível lógico presente na entrada D quando uma transição positiva ocorre.

EXEMPLO 5-7

Como um flip-flop J-K pode ser modificado para operar como um flip-flop D?

Solução

Esta modificação pode ser vista na Fig. 5-26, e é a mesma que foi feita para o caso do flip-flop S-C na Fig. 5-25.

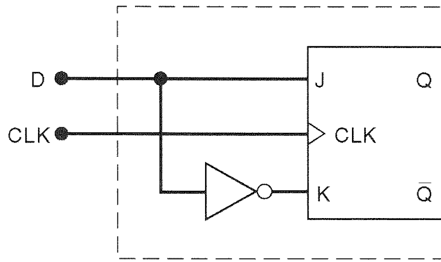


Fig. 5-26 Implementação de um flip-flop D disparado por transição a partir de um flip-flop J-K.

Transferência de Dados em Paralelo

Neste ponto, você deve estar se perguntando sobre a utilidade do flip-flop D, uma vez que a saída Q parece ter o mesmo valor que a entrada D. Esta afirmação não está inteiramente correta, pois a saída Q assume o valor da entrada D em determinados instantes, e portanto não é idêntica a D (como exemplo, veja as formas de onda na Fig. 5-24).

Na maioria das aplicações do flip-flop D, a saída Q deve assumir valores de sua entrada D em determinados instantes de tempo precisamente definidos. Um exemplo disto pode ser observado na Fig. 5-27. As saídas X, Y, Z de um circuito combinacional devem ser transferidas para os FFs Q_1, Q_2 e Q_3 para armazenamento. Utilizando-se flip-flops D, os níveis lógicos presentes em X, Y e Z são transferidos, respectivamente, para Q_1, Q_2 e Q_3, mediante a aplicação de um pulso, chamado TRANSFER, nas entradas *CLK* dos flip-flops. Os FFs podem armazenar estes valores para serem processados posteriormente. Este é um exemplo de **transferência paralela** de dados binários, onde os bits X, Y e Z são transferidos *simultaneamente*.

Questões de Revisão

1. O que acontecerá com a forma de onda da saída Q na Fig. 5-24(b) se a entrada D for mantida permanentemente em nível BAIXO?
2. *Falso* ou *verdadeiro*: A saída Q será igual ao nível lógico presente na entrada D em todos os instantes.
3. Flip-flops J-K podem ser utilizados para transferência de dados em paralelo?

5-8 LATCH *D* (LATCH TRANSPARENTE)

O flip-flop disparado por transição utiliza um detector de transição para assegurar que a saída vai responder à entrada *somente* quando uma transição de disparo do sinal de clock ocorrer. Se este detector não for usado, o circuito resultante vai operar de um modo diferente. Este circuito é chamado de **latch *D*** e seu circuito pode ser visto na Fig. 5-28(a).

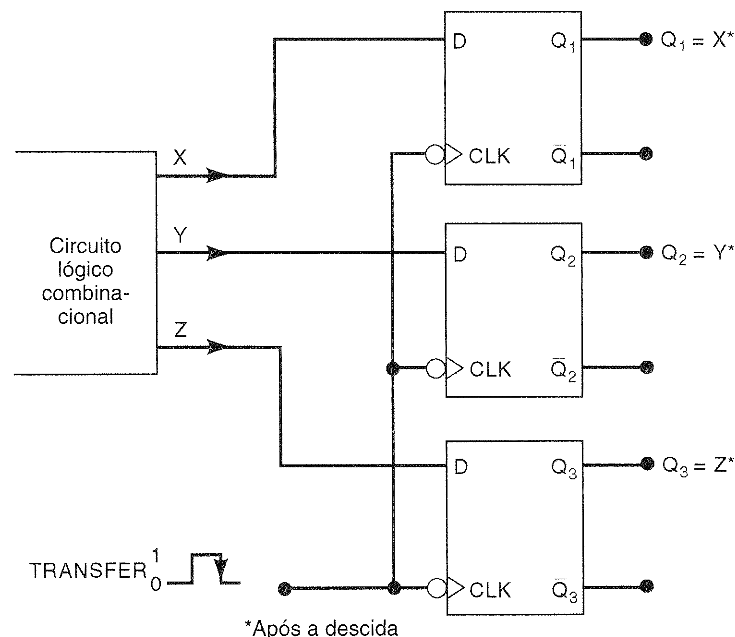
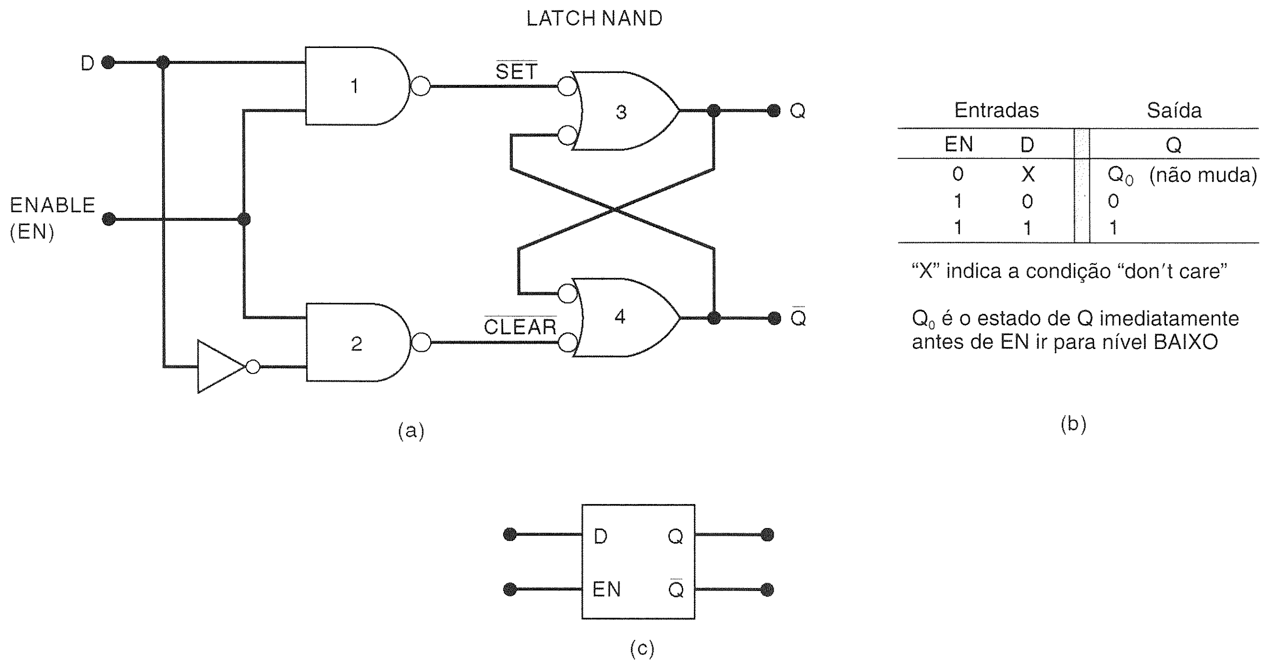


Fig. 5-27 Transferência de dados binários em paralelo usando flip-flops D.



Entradas		Saída
EN	D	Q
0	X	Q_0 (não muda)
1	0	0
1	1	1

"X" indica a condição "don't care"

Q_0 é o estado de Q imediatamente antes de EN ir para nível BAIXO

Fig. 5-28 Latch *D*: (a) estrutura; (b) tabela-verdade; (c) símbolo lógico.

O circuito contém um latch NAND e um circuito direcionador formado pelas portas NAND 1 e 2, *mas* não possui o detector de transição. A entrada em comum das portas direcionadoras é chamada de entrada de *habilitação* (do inglês *enable*, abreviando-se *EN*) em vez de ser chamada de entrada de clock, uma vez que seu efeito sobre as saídas *Q* e $\overline{Q}$ não está restrito às transições. A operação do latch D é descrita a seguir:

1. Quando *EN* está em ALTO, a entrada *D* vai produzir um nível BAIXO ou na entrada $\overline{SET}$ ou na entrada $\overline{CLEAR}$ do latch formados pelas portas NAND 3 e 4. Isto faz com que a saída *Q* fique no mesmo nível lógico que a entrada *D*. Se *D* mudar de estado enquanto *EN* estiver ALTO, *Q* acompanhará estas mudanças. Em outras palavras, enquanto *EN* = 1, a saída *Q* será igual à entrada *D*. Neste modo, diz-se que o latch é "transparente".
2. Quando *EN* vai para o nível BAIXO, a entrada *D* é impedida de alterar o estado do latch NAND, uma vez que as saídas das portas direcionadoras estão ambas em ALTO. Logo, as saídas *Q* e $\overline{Q}$ permanecerão no mesmo nível lógico em que estavam imediatamente antes de *EN* ir para o nível BAIXO. Em outras palavras, o valor das saídas está "fixo" neste nível, e não pode mudar de valor enquanto *EN* estiver em BAIXO, mesmo que *D* mude seu valor.

Esta operação está resumida na tabela-verdade da Fig. 5-28(b). O símbolo lógico para o latch *D* é mostrado na Fig. 5-28(c). Note que, apesar de a entrada *EN* operar de modo semelhante à entrada *CLK* em flip-flops disparados por transição, não existe o pequeno triângulo na entrada *EN*. Isto acontece porque o pequeno triângulo é usado estritamente para indicar entradas que podem causar mudanças na saída apenas quando uma transição ocorre. *O latch D não é disparado por transição.*

EXEMPLO 5-8

Determine a forma de onda para a saída *Q* do latch *D* com as formas de onda das entradas *EN* e *D* mostradas na Fig. 5-29. Suponha que inicialmente *Q* = 0.

Solução

Anteriormente ao instante T_1, *EN* está em BAIXO, e portanto a saída *Q* é mantida em nível 0 e não pode mudar de estado, mesmo que *D* mude. Durante o intervalo T_1 até T_2, *EN* está em ALTO, e assim a saída *Q* acompanha o sinal presente em *D*. Logo, *Q* vai para ALTO em T_1 e permanece neste estado porque *D* também não se altera. Quando *EN* volta a BAIXO em T_2, *Q* vai permanecer em ALTO, uma vez que era este nível lógico que estava em *D* no instante T_2, e permanece neste estado enquanto *EN* estiver em BAIXO.

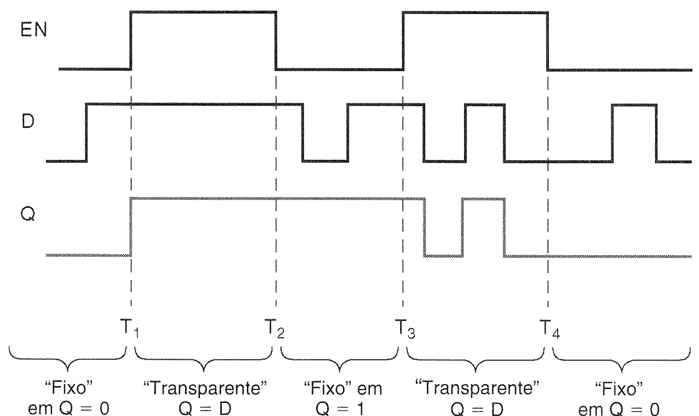


Fig. 5-29 Formas de onda do Exemplo 5-8 mostrando os dois modos de operação do latch *D* transparente.

No instante T_3, *EN* vai novamente para ALTO, fazendo com que a saída *Q* acompanhe as mudanças na entrada *D* até T_4, quando *EN* retorna a BAIXO. Durante o intervalo entre T_3 e T_4, o latch *D* está "transparente", uma vez que as variações na entrada *D* se propagam para a saída *Q*. Em T_4, quando *EN* vai para BAIXO, *Q* permanece em BAIXO, pois era este o nível presente em *D* no instante T_4. Após T_4, as variações em *D* não afetam *Q*, uma vez que a saída está "fixa", pois *EN* = 0.

Questões de Revisão

1. Descreva como a operação de um latch *D* é diferente da operação de um flip-flop disparado por transição.
2. *Falso* ou *verdadeiro*: Um latch *D* é "transparente" quando *EN* = 0.
3. *Falso* ou *verdadeiro*: Em um latch *D*, a entrada *D* pode afetar a saída apenas quando *EN* = 1.

5-9 ENTRADAS ASSÍNCRONAS

Para os flip-flops com clock que estamos estudando, as entradas *S*, *C*, *J*, *K* e *D* são chamadas de entradas de *controle*. Elas também são chamadas de *entradas síncronas*, porque seu efeito sobre a saída é sincronizado com a entrada *CLK*. Como já vimos, as entradas de controle síncronas devem ser usadas em conjunto com o sinal de clock para disparar o flip-flop.

A maioria dos flip-flops com clock também possui uma ou mais **entradas assíncronas** que operam independentemente das entradas síncronas e da entrada de clock. Estas entradas assíncronas podem ser usadas para colocar o flip-flop no estado 0 ou no estado 1, *em qualquer instante, independentemente das condições das outras entradas*. Em outras palavras, as entradas assíncronas são chamadas de **entradas de sobreposição**, pois se sobrepõem a todas as outras entradas para colocar o flip-flop em um determinado estado.

A Fig. 5-30 mostra um flip-flop J-K com duas entradas assíncronas identificadas como $\overline{\text{PRESET}}$ e $\overline{\text{CLEAR}}$. Estas entradas são ativas em BAIXO, conforme indicam as bolhas presentes no símbolo do flip-flop. A tabela-verdade resume como estas entradas afetam a saída do flip-flop. Vamos examinar as várias possibilidades:

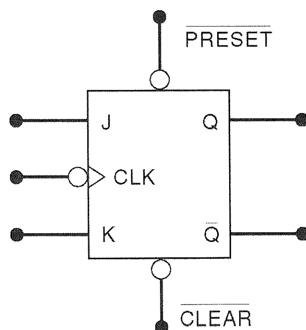
- **$\overline{\text{PRESET}}$ = $\overline{\text{CLEAR}}$ = 1**. As entradas assíncronas estão inativas e o flip-flop está livre para responder às entradas *J*, *K* e *CLK*, ou seja a operação síncrona pode ser realizada.
- **$\overline{\text{PRESET}}$ = 0; $\overline{\text{CLEAR}}$ = 1.** Como a entrada $\overline{\text{PRESET}}$ está ativa, *Q* é *imediatamente* colocado em 1, quaisquer que sejam os níveis presentes nas entradas *J*, *K* e *CLK*. A entrada *CLK* não pode afetar o flip-flop enquanto $\overline{\text{PRESET}}$ = 0.
- **$\overline{\text{PRESET}}$ = 1; $\overline{\text{CLEAR}}$ = 0.** Como a entrada $\overline{\text{CLEAR}}$ está ativa, *Q* é *imediatamente* limpo (*Q* = 0), quaisquer que sejam os níveis presentes nas entradas *J*, *K* e *CLK*. A entrada *CLK* não pode afetar o flip-flop enquanto $\overline{\text{CLEAR}}$ = 0.
- **$\overline{\text{PRESET}}$ = $\overline{\text{CLEAR}}$ = 0.** Esta condição não deve ser usada, pois pode resultar em uma resposta ambígua.

É importante perceber que essas entradas assíncronas respondem a níveis de tensão contínua (DC). Isto significa que se um nível 0 for mantido na entrada $\overline{\text{PRESET}}$, a saída *Q* permanecerá no estado *Q* = 1, independentemente do que estiver ocorrendo com as outras entradas. Do mesmo modo, um nível BAIXO constante em $\overline{\text{CLEAR}}$ mantém o flip-flop no estado *Q* = 0. Então, podemos dizer que as entradas assíncronas podem ser utilizadas para manter o flip-flop em um determinado estado pelo tempo que desejarmos. Na maioria das vezes, entretanto, as entradas assíncronas são usadas para colocar o flip-flop no estado desejado através da aplicação de um pulso momentâneo.

Muitos dos flip-flops com clock que estão disponíveis em circuitos integrados possuem as duas entradas assíncronas, enquanto outros possuem apenas a entrada $\overline{\text{CLEAR}}$. Alguns flip-flops possuem as entradas assíncronas ativas em ALTO, em vez de ativas em BAIXO. Para estes casos, o símbolo do flip-flop não possui a bolha de inversão nas entradas assíncronas.

Designações para as Entradas Assíncronas

Os fabricantes de circuitos integrados ainda não concordaram quanto à nomenclatura a ser utilizada para as entradas assíncronas. As denominações mais comuns são *PRE* (abreviatura de PRESET) e *CLR* (abreviatura de CLEAR). As designações S_D (SET direto) e R_D (RESET direto) também são usadas. De agora em diante, usaremos as designações *PRE* e *CLR* para indicar as entradas assíncronas, uma vez que estas são as designações mais usadas. Quando estas entradas assíncronas são ativas em BAIXO, como geralmente são,



$\overline{\text{PRESET}}$	$\overline{\text{CLEAR}}$	Resposta do FF
1	1	Operação com clock*
0	1	Q = 1
1	0	Q = 0
0	0	Não usada

*Q irá responder a J, K e CLK

Fig. 5-30 Flip-flop J-K com clock e entradas assíncronas.

usaremos uma barra sobreposta para indicar esta condição, isto é, $\overline{PRE}$ e $\overline{CLR}$.

Embora a maioria do CIs de flip-flops possua pelo menos uma ou mais entradas assíncronas, existem algumas aplicações nas quais elas não são usadas. Neste caso, elas devem ser mantidas permanentemente em seu nível inativo. Muitas vezes durante o uso de flip-flops no restante do texto não mostraremos as entradas assíncronas não-utilizadas, e consideraremos que elas estão permanentemente conectadas ao seu nível lógico inativo.

EXEMPLO 5-9

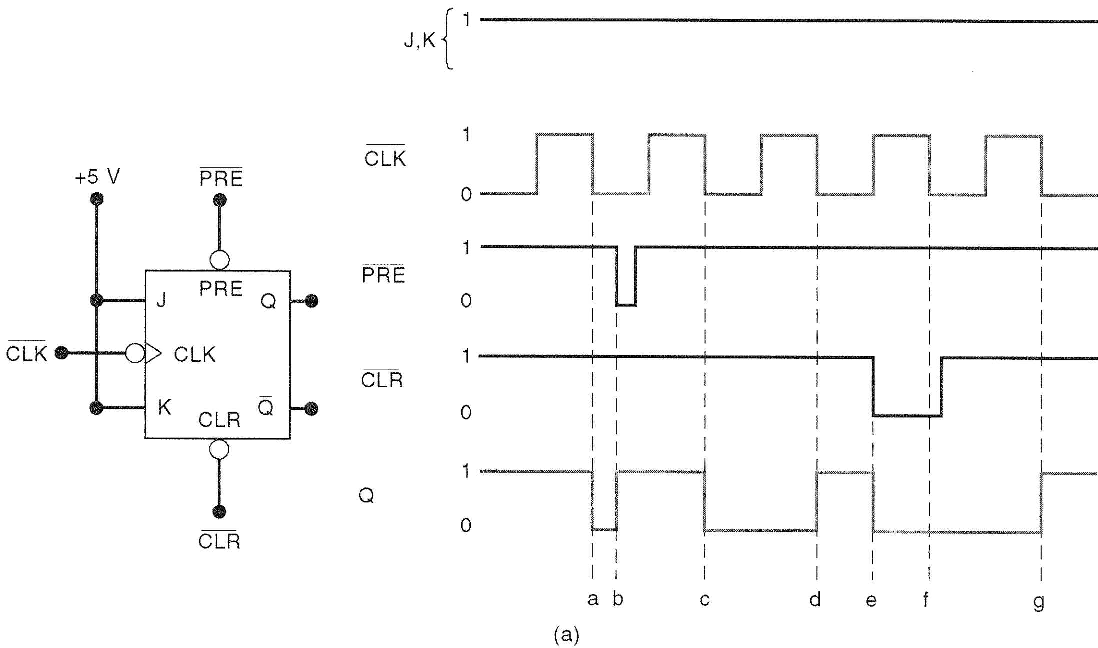
A Fig. 5-31(a) mostra o símbolo de um flip-flop J-K que é disparado por transições negativas do sinal que está na entrada *CLK* e que possui entradas assíncronas ativas em BAIXO. Antes de prosseguir com este exemplo, observe o modo pelo qual as entradas são denominadas. Primeiro, note que o sinal de clock aplicado ao flip-flop é denominado de $\overline{CLK}$ (a barra sobreposta indica que o sinal é ativo na transição negativa), enquanto no outro lado da bolha, dentro do bloco, ele é denominado *CLK*. Do mesmo modo, as entradas assíncronas externas, ativas em BAIXO, são denominadas $\overline{PRE}$ e $\overline{CLR}$, enquanto dentro do bloco, do outro lado da bolha, são denominadas *PRE* e *CLR*. O mais importante a ser lembrado é que a presença da bolha na entrada significa que esta responde a um nível lógico BAIXO.

As entradas *J* e *K* estão conectadas em ALTO neste exemplo. Determine a saída *Q* em função das formas de onda de entrada mostradas na Fig. 5-31(a). Suponha que a saída *Q* está inicialmente em ALTO.

Solução

Inicialmente, $\overline{PRE}$ e $\overline{CLR}$ estão em seu estado inativo ALTO, e portanto eles não terão efeito sobre *Q*. Quando a primeira transição negativa do sinal $\overline{CLK}$ ocorre no ponto *a*, *Q* vai comutar para seu estado oposto (lembre-se de que $J = K = 1$ causa a troca de estado).

No ponto *b*, a entrada $\overline{PRE}$ é pulsada em BAIXO. Isto faz com que o flip-flop seja *imediatamente* colocado no estado $Q = 1$. Observe que $\overline{PRE}$ faz $Q = 1$ sem esperar por



Ponto	Operação
a	Comutação sincronizada com a descida de $\overline{CLK}$
b	Q é assincronamente colocado em 1 quando $\overline{PRE} = 0$
c	Comutação síncrona
d	Comutação síncrona
e	Q é assincronamente colocado em 0 quando $\overline{CLR} = 0$
f	$\overline{CLR}$ se sobrepõe à transição negativa de $\overline{CLK}$
g	Comutação síncrona

(b)

Fig. 5-31 Formas de onda do Exemplo 5-9 mostrando como um flip-flop J-K com clock responde às entradas assíncronas.

uma transição negativa de $\overline{CLK}$. As entradas assíncronas operam independentemente de $\overline{CLK}$.

No ponto *c*, a transição negativa de $\overline{CLK}$ vai novamente fazer com que a saída *Q* mude para seu estado oposto. Observe que $\overline{PRE}$ retornou a seu estado inativo antes do ponto *c*. Do mesmo modo, a transição negativa do sinal $\overline{CLK}$ no ponto *d* vai fazer com que a saída *Q* volte a um nível ALTO.

No ponto *e*, a entrada $\overline{CLR}$ é pulsada em BAIXO e vai fazer *imediatamente* com que a saída *Q* seja igual a 0. Mais uma vez, isto acontece independentemente de $\overline{CLK}$.

Uma transição negativa de $\overline{CLK}$ no ponto *f não vai comutar Q*, porque a entrada $\overline{CLR}$ ainda está ativa. Um nível BAIXO em $\overline{CLR}$ se sobrepõe à entrada $\overline{CLK}$ e mantém $Q = 0$.

Quando ocorre uma transição negativa de $\overline{CLK}$ no ponto *g*, isto faz com que a saída *Q* vá para o estado ALTO, uma vez que nenhuma das entradas assíncronas está ativa neste ponto.

Estes passos estão resumidos na Fig. 5-31(b).

Questões de Revisão

1. De que modo o funcionamento de uma entrada assíncrona difere da operação de uma entrada síncrona?
2. Um flip-flop D pode responder às suas entradas *D* e *CLK* enquanto $\overline{PRE} = 1$?
3. Enumere as condições necessárias para que um flip-flop J-K disparado por transição positiva e com entradas assíncronas ativas em BAIXO comute para seu estado oposto.

5-10 SÍMBOLOS IEEE/ANSI

Utilizamos os símbolos tradicionais para cada um dos latches e flip-flops que estudamos até o momento, e continuaremos a usar estes símbolos na maioria dos nossos diagramas de circuitos. Nesta seção, examinaremos os símbolos IEEE/ANSI para estes mesmos dispositivos a fim de que você fique familiarizado com eles.

A Fig. 5-32(a) mostra o símbolo lógico para um latch *D*. Este é o símbolo IEEE/ANSI. Ele utiliza a letra "C" para designar a entrada ENABLE. Como veremos, a simbologia IEEE/ANSI utiliza a letra "C" para qualquer entrada que *controle* quando outras entradas terão efeito ou não sobre a saída. Como sabemos, o nível lógico aplicado na entrada ENABLE controla quando deve ser permitido que a entrada *D* altere o estado de *Q* e $\overline{Q}$. Observe que os símbolos *Q* e $\overline{Q}$ estão colocados do lado de fora do bloco, e observe também o triângulo em $\overline{Q}$ para indicar que esta é uma saída invertida. Este é o padrão para a simbologia IEEE/ANSI. Lembre-se de que este triângulo tem a mesma função das bolhas de inversão usadas nos símbolos mais antigos.

A Fig. 5-32(b) mostra o símbolo IEEE/ANSI para um CI específico, o latch quádruplo TTL 74LS375. Este CI contém quatro latches *D* que operam individualmente, do mesmo modo que descrevemos anteriormente. Este símbolo também se aplica aos CIs correspondentes em outras séries TTL e CMOS, como por exemplo o 74HC375.

Se examinarmos o símbolo lógico para este CI, podemos destacar vários pontos importantes. Em primeiro lugar, podemos notar que o símbolo contém quatro retângulos menores que representam os latches individuais. Observe também como é feita a indicação das entradas e saídas em cada latch. Por exemplo, a entrada *D* do latch superior é identificada como "1*D*", sua entrada de habilitação é chamada "*C*1" e suas saídas são denominadas 1*Q* e 1$\overline{Q}$. Finalmente, observe que os dois latches superiores possuem a entrada de habilitação em comum, isto é, *C*1 e *C*2 estão conectadas internamente a um mesmo ponto, que por sua vez está ligado a um único pino do CI. Do mesmo modo, os dois latches inferiores compartilham uma mesma entrada de habilitação.

A Fig. 5-33(a) mostra o símbolo IEEE/ANSI para um flip-flop J-K disparado por transição negativa e com entradas assíncronas. A entrada de clock é denominada "C" no interior do símbolo. Observe também que existem dois triângulos na entrada de clock. O primeiro está localizado no interior do bloco e indica que esta entrada é disparada por transição. O outro que está do lado de fora indica que o

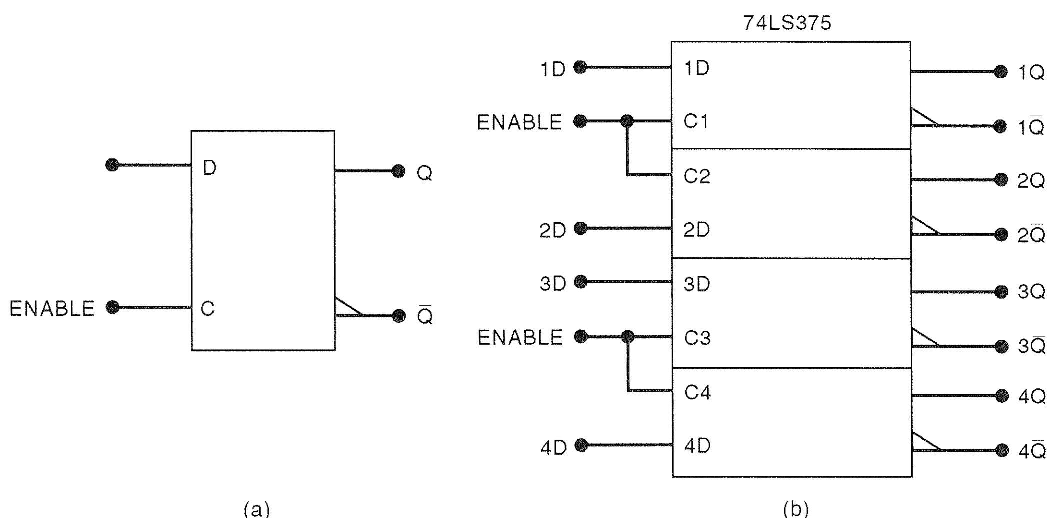


Fig. 5-32 Símbolos IEEE/ANSI para: (a) um latch *D* e (b) CI 74LS375, latch quádruplo.

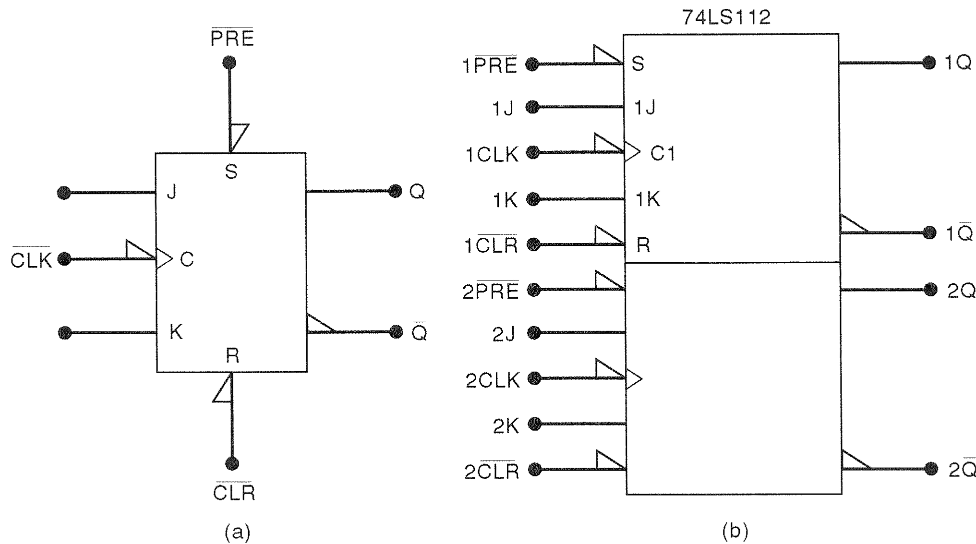


Fig. 5-33 Símbolos IEEE/ANSI para: (a) um flip-flop J-K disparado por transição e (b) CI 74LS112, flip-flop J-K duplo disparado por transição negativa.

flip-flop é disparado na transição negativa. As entradas $\overline{PRE}$ e $\overline{CLR}$ são ativas em BAIXO, como está simbolizado pelos triângulos retos. É interessante observar que a simbologia IEEE/ANSI utiliza as denominações "S" e "R" no interior do símbolo para indicar as entradas assíncronas SET e RESET, que são equivalentes a PRESET e CLEAR, respectivamente.

A Fig. 5-33(b) mostra o símbolo IEEE/ANSI para um CI que faz parte da série 74LS da família TTL. O 74LS112 é um flip-flop J-K duplo disparado por transição negativa com entradas assíncronas. Ele contém dois flip-flops como o que pode ser visto na Fig. 5-33(a). Observe como as entradas e saídas estão numeradas. Observe também que as denominações das entradas dentro dos retângulos aparecem apenas no flip-flop superior. Fica subentendido que o flip-flop na parte inferior segue esta mesma disposição. Este mesmo símbolo se aplica ao CI 74HC112 da família CMOS.

A Fig. 5-34(a) mostra o símbolo IEEE/ANSI para um flip-flop *D* disparado na subida e com entradas assíncronas. Observe que não existe o triângulo reto na entrada de clock, uma vez que o flip-flop é disparado na transição positiva.

A Fig. 5-34(b) mostra o símbolo IEEE/ANSI para o CI 74HC175, que contém quatro flip-flops que compartilham uma entrada *CLK* e uma entrada $\overline{CLR}$. Os flip-flops não têm a entrada $\overline{PRE}$. Este símbolo contém retângulos separados para representar cada um dos flip-flops, e um **bloco de controle comum**, que é representado pelo retângulo com

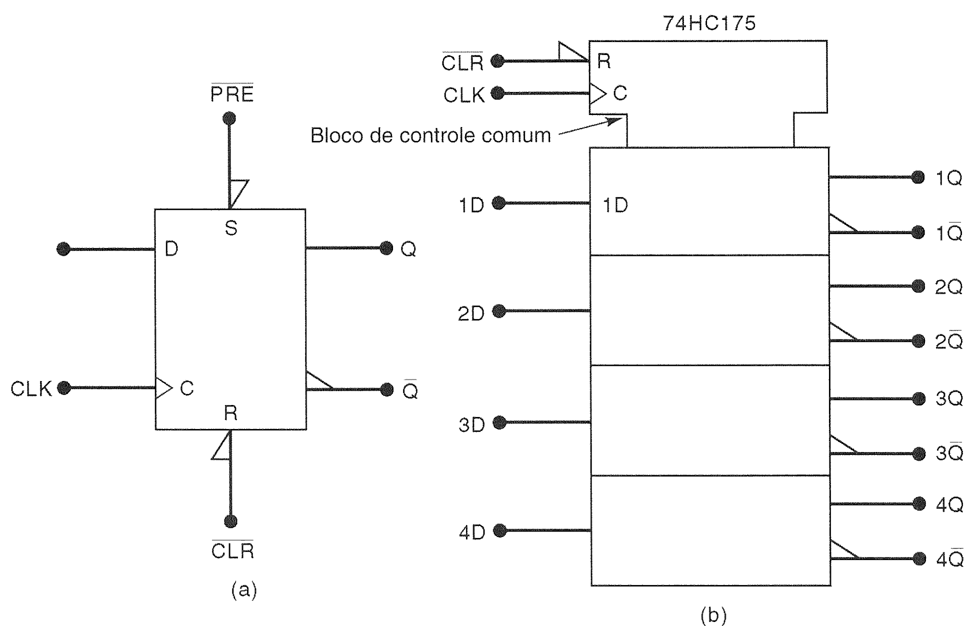


Fig. 5-34 Símbolos IEEE/ANSI para: (a) um flip-flop D disparado por transição e (b) CI 74HC175, flip-flop quádruplo com entradas comuns de clock e clear.

chanfros na parte superior do símbolo. Este bloco de controle comum é usado sempre que um CI possui uma ou mais entradas que são comuns a mais de um circuito no chip. No caso do 74HC175, as entradas CLK e $\overline{CLR}$ são comuns a todos os flip-flops D existentes no CI. Isto significa que uma transição positiva na entrada CLK faz cada saída Q assumir o nível presente na sua respectiva entrada D. Isto também significa que um nível BAIXO em $\overline{CLR}$ coloca todas as saídas Q em BAIXO.

Questões de Revisão

1. Explique o significado dos dois tipos de triângulos que podem fazer parte da simbologia IEEE/ANSI em uma entrada de clock.
2. Descreva o significado do bloco de controle comum.

5-11 CONSIDERAÇÕES SOBRE TEMPORIZAÇÃO EM FLIP-FLOPS

Os fabricantes de CIs de flip-flops especificam muitos parâmetros importantes de temporização e características que devem ser considerados antes que um FF possa ser usado em um circuito. Descreveremos os mais importantes e apresentaremos alguns exemplos reais de CIs de flip-flops das famílias lógicas TTL e CMOS.

Tempos de Setup e Hold

Os tempos de setup e hold já foram discutidos, e você deve lembrar da Seção 5-4 que eles representam restrições que devem ser satisfeitas para disparar confiavelmente um FF. A folha de características do fabricante do CI sempre especifica os valores *mínimos* de t_S e t_H.

Atrasos de Propagação

Sempre que um sinal causa a mudança de estado da saída de um FF, existe um atraso entre a aplicação do sinal e o momento em que a saída muda. A Fig. 5-35 ilustra os **atrasos de propagação** que ocorrem em resposta a uma transição positiva na entrada CLK. Repare que estes atrasos são medidos entre os pontos de 50% de amplitude das formas de onda de entrada e saída. Os mesmos tipos de atrasos acontecem em resposta a sinais aplicados nas entradas assíncronas (PRESET e CLEAR). As folhas de características dos fabricantes usualmente especificam os valores *máximos* para t_{PLH} e t_{PHL}.*

Os modernos CIs de flip-flops possuem atrasos de propagação que variam de uns poucos nanossegundos até por volta de 100 ns. Os valores de t_{PLH} e t_{PHL} geralmente não são os mesmos, e eles aumentam de modo diretamente proporcional ao número de cargas sendo acionadas pela saída Q. Os atrasos de propagação dos FFs têm um importante papel em determinadas situações que encontraremos mais adiante.

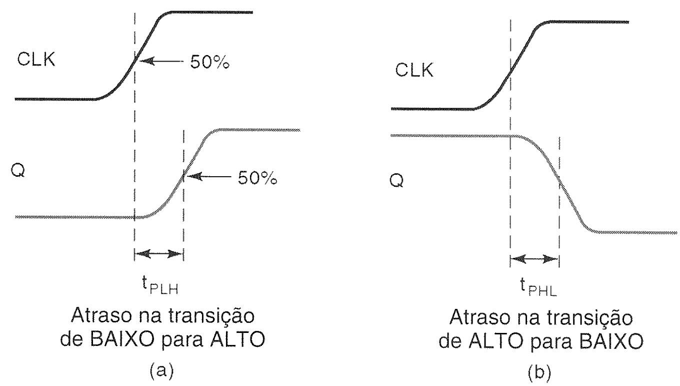


Fig. 5-35 Atrasos de propagação nos FFs.

Freqüência Máxima de Clock, f_{MAX}

Esta é a freqüência mais alta que pode ser aplicada na entrada CLK de um FF e ainda dispará-lo confiavelmente. O limite f_{MAX} varia de FF para FF, mesmo entre os FFs que têm o mesmo número. Por exemplo, o fabricante do CI 7470 flip-flop J-K testa vários destes FFs e pode constatar que os valores para f_{MAX} ficam na faixa de 20 a 35 MHz. Ele então especifica a f_{MAX} *mínima* como 20 MHz. Isto pode parecer confuso, mas um pouco de raciocínio deve tornar claro que o fabricante está informando que ele não pode garantir que o FF 7470, que você vai usar no seu circuito, vai operar acima de 20 MHz; a maioria deles funcionará acima disto, mas alguns deles não. Entretanto, se o circuito operar abaixo de 20 MHz, ele garante que os FFs funcionarão corretamente.

Tempos de Duração em ALTO e BAIXO do Sinal de Clock

O fabricante também especifica o tempo de duração *mínimo* que o sinal de CLK deve permanecer em BAIXO antes de ir para ALTO, algumas vezes denominado $t_W(L)$, e o tempo mínimo que CLK deve ser mantido ALTO antes de retornar para BAIXO, algumas vezes chamado $t_W(H)$. Estes tempos são definidos na Fig. 5-36(a). O não-atendimento a estes requisitos de tempos mínimos pode resultar em disparos não-confiáveis. Note que estes valores de tempo são medidos entre os pontos a meio caminho das transições do sinal.

Largura dos Pulsos Assíncronos

O fabricante também especifica o tempo de duração *mínimo* que as entradas PRESET e CLEAR devem ser mantidas no seu estado ativo, de modo a setar ou ressetar confiavelmente o flip-flop. A Fig. 5-36(b) mostra $t_W(L)$ para entradas assíncronas ativas em BAIXO.

*t_P se refere a tempo de propagação, e as letras L e H são as iniciais de LOW (BAIXO) e HIGH (ALTO), logo t_{PLH} = tempo de propagação de LOW (BAIXO) para HIGH (ALTO), e t_{PHL} = tempo de propagação de HIGH (ALTO) para LOW (BAIXO). (*N. do T.*)

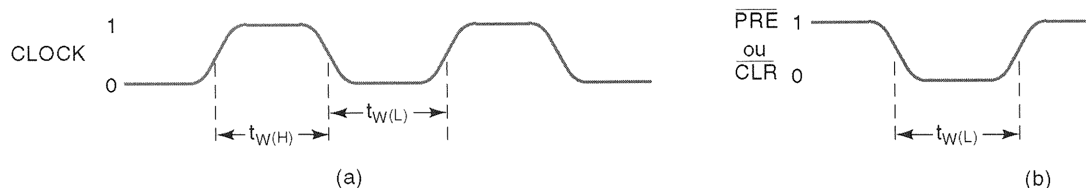


Fig. 5-36 (a) Tempos de duração do clock em BAIXO e em ALTO; (b) largura de pulso assíncrono.

Tempos de Transição do Clock

Para garantir um disparo confiável, os tempos de transição da forma de onda do clock (tempos de subida e descida) devem ser mantidos bem pequenos. Se o sinal de clock demorar muito para fazer sua transição de um nível para o outro, o FF pode disparar de modo errático ou simplesmente não disparar. Os fabricantes usualmente não relacionam os requisitos de tempos de transição máximos para cada circuito integrado de FF. Em vez disso, normalmente isto é dado como um requisito geral para todos os CIs de uma determinada família lógica. Por exemplo, os tempos de transição são geralmente ≤ 50 ns para dispositivos TTL e ≤ 200 ns para CMOS. Estes requisitos variam entre os diferentes fabricantes e entre as diversas subfamílias lógicas TTL e CMOS.

CIs Reais

Como exemplos práticos desses parâmetros de temporização, vamos dar uma olhada em vários circuitos integrados reais de FFs. Em particular, analisaremos os seguintes CIs:

- **7474** Duplo flip-flop D disparado pela borda (TTL padrão)
- **74LS112** Duplo flip-flop J-K disparado pela borda (TTL Schottky de baixa potência)
- **74C74** Duplo flip-flop D disparado pela borda (CMOS de porta metálica)
- **74HC112** Duplo flip-flop J-K disparado pela borda (CMOS de alta velocidade)

A Tabela 5-2 relaciona diversos parâmetros de temporização para cada um destes FFs, conforme apresentados nos manuais dos fabricantes. Todos os valores relacionados são valores *mínimos*, exceto para os atrasos de propagação, que são valores *máximos*. Um exame da Tabela 5-2 revela dois aspectos interessantes.

1. Todos os FFs têm um t_H muito baixo; isto é típico na maioria dos modernos FFs disparados por transição.
2. A série 74HC de dispositivos CMOS tem valores de temporização comparáveis aos dos dispositivos TTL. A série 74C é muito mais lenta do que a série 74HC.

EXEMPLO 5-10

Com referência à Tabela 5-2, determine o seguinte:

(a) Considere que a saída $Q = 0$. Quanto tempo demora para Q ir para ALTO quando uma transição positiva ocorre na entrada *CLK* de um 7474?

(b) Suponha que a saída $Q = 1$. Quanto tempo demora para Q ir para BAIXO em resposta à entrada $\overline{CLR}$ de um 74HC112?

(c) Qual é o pulso mais estreito que pode ser aplicado na entrada $\overline{CLR}$ de um FF 74LS112 para limpar a saída Q de modo confiável?

(d) Qual dos FFs na Tabela 5-2 necessita que as entradas de controle permaneçam estáveis *depois* da ocorrência da transição ativa do clock?

(e) Para quais FFs as entradas de controle devem ser mantidas estáveis, por um certo tempo mínimo, antes da transição ativa do clock?

Solução

(a) A transição positiva faz Q ir de BAIXO para ALTO. O atraso do *CLK* para a saída Q é relacionado como t_{PLH} = 25 ns para o 7474.

TABELA 5-2 Parâmetros de temporização de flip-flops (em nanossegundos)

		TTL		CMOS	
		7474	74LS112	74C74	74HC112
t_S		20	20	60	25
t_H		5	0	0	0
t_{PHL}	de *CLK* para *Q*	40	24	200	31
t_{PLH}	de *CLK* para *Q*	25	16	200	31
t_{PHL}	de $\overline{CLR}$ para *Q*	40	24	225	41
t_{PLH}	de $\overline{PRE}$ para *Q*	25	16	225	41
$t_W(L)$	tempo em BAIXO de *CLK*	37	15	100	25
$t_W(H)$	tempo em ALTO de *CLK*	30	20	100	25
$t_W(L)$	para $\overline{PRE}$ ou $\overline{CLR}$	30	15	60	25
f_{MAX}	*em MHz*	15	30	5	20

(b) Para o 74HC112, o tempo necessário para Q ir de ALTO para BAIXO em resposta à entrada $\overline{CLR}$ é apresentado como $t_{PHL} = 41$ ns.
(c) Para o 74LS112, o pulso mais estreito para a entrada $\overline{CLR}$ é relacionado como $t_W(L) = 15$ ns.
(d) O 7474 é o único FF da Tabela 5-2 que tem um tempo de hold diferente de zero.
(e) Todos os FFs têm o requisito de tempo de setup diferente de zero.

Questões de Revisão

1. Quais parâmetros de temporização dos FFs indicam o tempo que a saída Q leva para responder a uma entrada?
2. *Verdadeiro ou falso:* Um FF que tem f_{MAX} de 25 MHz pode ser disparado confiavelmente por qualquer forma de onda pulsada em *CLK* com uma freqüência inferior a 25 MHz.

5-12 PROBLEMAS POTENCIAIS DE TEMPORIZAÇÃO EM CIRCUITOS COM FLIP-FLOPS

Em muitos circuitos digitais, a saída de um FF é conectada, diretamente ou através de portas lógicas, na entrada de outro FF, e ambos os FFs são disparados pelo mesmo sinal de clock. Isto representa um problema potencial de temporização. Uma situação típica está ilustrada na Fig. 5-37, onde a saída de Q_1 está conectada na entrada J de Q_2 e ambos os FFs são disparados por suas entradas *CLK* pelo mesmo sinal.

O problema potencial de temporização é este: como Q_1 muda na descida do pulso de clock, a entrada J_2 de Q_2 estará mudando quando ele receber a mesma transição negativa. Isto pode levar a uma resposta imprevisível de Q_2.

Vamos supor que inicialmente $Q_1 = 1$ e $Q_2 = 0$. Logo, o FF Q_1 tem $J_1 = K_1 = 1$, e Q_2 tem $J_2 = Q_1 = 1$ e $K_2 = 0$, antes da descida do pulso de clock. Quando a transição negativa ocorre, Q_1 comutará para o estado BAIXO, mas só vai realmente para BAIXO após o atraso de propagação, t_{PHL}. A mesma transição negativa vai disparar confiavelmente Q_2 para o estado ALTO desde que o t_{PHL} seja maior do que o tempo de hold de Q_2, t_H. Se esta condição não for satisfeita, a resposta de Q_2 será imprevisível.

Felizmente, todos os FFs disparados por transição mais recentes possuem tempo de hold de 5 ns ou menos, e a maioria tem $t_H = 0$, o que significa que eles não têm a restrição do tempo de hold. Para estes FFs, as situações como a apresentada na Fig. 5-37 não representam um problema.

A menos que seja informado o contrário, em todos os circuitos com FFs que encontraremos neste livro, presumiremos que o requisito de tempo de hold do FF é pequeno o suficiente para que ele responda de maneira confiável, conforme a seguinte regra:

A saída do FF vai para o estado determinado pelos níveis lógicos presentes em suas entradas de controle síncronas imediatamente antes da transição de disparo do clock.

Se aplicamos esta regra à Fig. 5-37, obtemos que a saída Q_2 vai para um estado determinado pela condição $J_2 = 1$, $K_2 = 0$, que está presente imediatamente antes da transição nega-

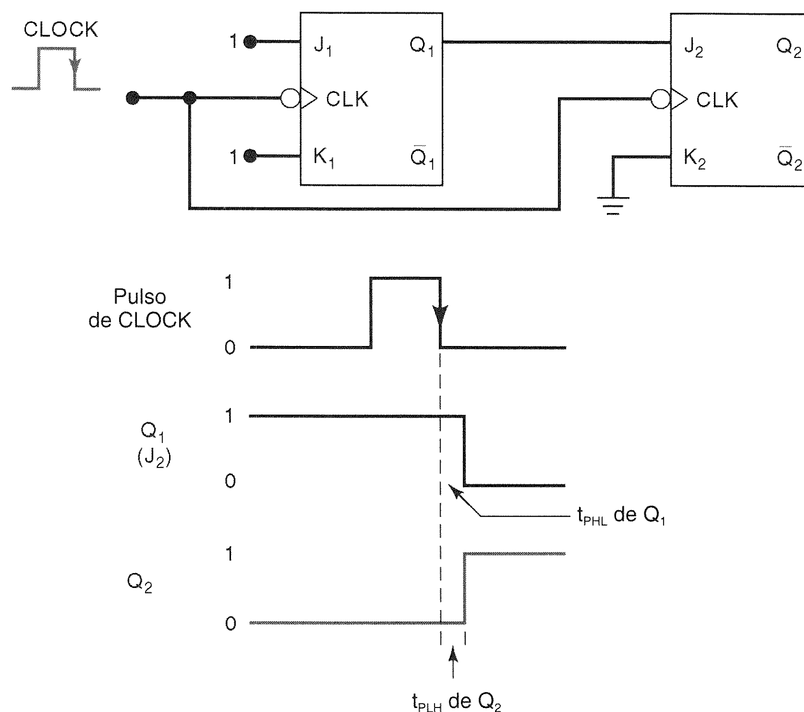


Fig. 5-37 Q_2 responderá adequadamente ao nível presente em Q_1 antes da descida do clock, desde que o tempo de hold de Q_2, t_H, seja menor do que o atraso de propagação de Q_1.

tiva do pulso de clock. O fato de J_2 estar mudando em resposta à mesma transição de descida não tem efeito algum.

EXEMPLO 5-11

Determine a saída Q para o flip-flop J-K disparado pela descida, para as formas de onda de entrada da Fig. 5-38. Suponha que $t_H = 0$ e que inicialmente $Q = 0$.

Solução

O FF responderá apenas nos instantes de tempo T_2, T_4, T_6 e T_8. Em T_2, Q responde à condição $J = K = 0$ presente na entrada imediatamente antes de T_2. Em T_4, Q responde à condição $J = 1$, $K = 0$ presente na entrada imediatamente antes de T_4. Em T_6, Q responde à condição $J = 0$, $K = 1$ presente na entrada imediatamente antes de T_6. Em T_8, Q responde a $J = K = 1$.

5-13 FLIP-FLOPS MESTRE/ESCRAVO

Antes do desenvolvimento dos flip-flops disparados pela borda com requisito de tempo de hold muito pequeno ou nulo, problemas de temporização, tais como o da Fig. 5-37, eram freqüentemente tratados utilizando-se uma classe de FFs denominada **flip-flops mestre/escravo**. Um FF mestre/escravo na verdade contém dois FFs, um mestre e um escravo. Na subida do sinal *CLK*, os níveis nas entradas de controle (*D*, *J*, *K*) são usados para determinar a saída do mestre. Quando o sinal *CLK* vai para BAIXO, o estado do mestre é transferido para o escravo, cujas saídas são Q e $\overline{Q}$. Deste modo, Q e $\overline{Q}$ mudam logo após a descida do clock. Estes flip-flops do tipo mestre/escravo funcionam de modo similar aos FFs disparados por transição negativa, exceto por uma desvantagem importante: as entradas de controle devem ficar estáveis enquanto *CLK* está em ALTO, ou uma operação imprevisível pode ocorrer. Este problema com os FFs mestre/escravo foi resolvido com uma versão mestre/escravo melhorada denominada *mestre/escravo com travamento de dados* (*data lockout*).

O FF mestre/escravo tornou-se obsoleto, embora você possa encontrá-lo em equipamentos mais antigos. Exemplos deste tipo são os CIs TTL padrão 7473, 7476 e 74107, além das versões com travamento de dados 74110 e 74111. As novas tecnologias de CIs (74LS, 74AS, 74ALS, 74HC, 74HCT) não incluem FFs do tipo mestre/escravo nas suas séries. Na verdade, o 74LS76 e o 74LS107 são fabricados como FFs disparados pela borda, embora os componentes da série padrão sejam do tipo mestre/escravo.

Para a maioria dos casos, se você encontrar um FF mestre/escravo num equipamento, você pode analisá-lo como um FF disparado pela borda de descida.

5-14 APLICAÇÕES COM FLIP-FLOPS

No início do capítulo, apresentamos alguns exemplos de como os flip-flops com portas NAND e os flip-flops com portas NOR são usados para, respectivamente, eliminar o problema de trepidação de chave (Exemplo 5-2) e registro de evento (Exemplo 5-4). Estes simples FFs sem clock são de algum modo limitados em suas aplicações. FFs com clock oferecem ao projetista um grupo versátil de dispositivos que têm numerosas aplicações. Vamos introduzir superficialmente as aplicações mais comuns nas próximas seções, e nos aprofundaremos em capítulos subseqüentes.

5-15 SINCRONIZAÇÃO DE FLIP-FLOPS

A maioria dos sistemas digitais é predominantemente síncrona em sua operação, já que a maioria dos sinais muda de estado em sincronismo com as transições do clock. Em muitos casos, entretanto, existe um sinal externo que não é sincronizado com o clock; em outras palavras, ele é assíncrono. Sinais assíncronos ocorrem freqüentemente como resultado da atuação de um operador humano em uma chave de entrada em momentos aleatórios em relação ao sinal de clock. Esta ação randômica pode produzir resultados imprevisíveis e indesejáveis. O exemplo seguinte ilustra como um FF pode ser usado para sincronizar uma entrada assíncrona.

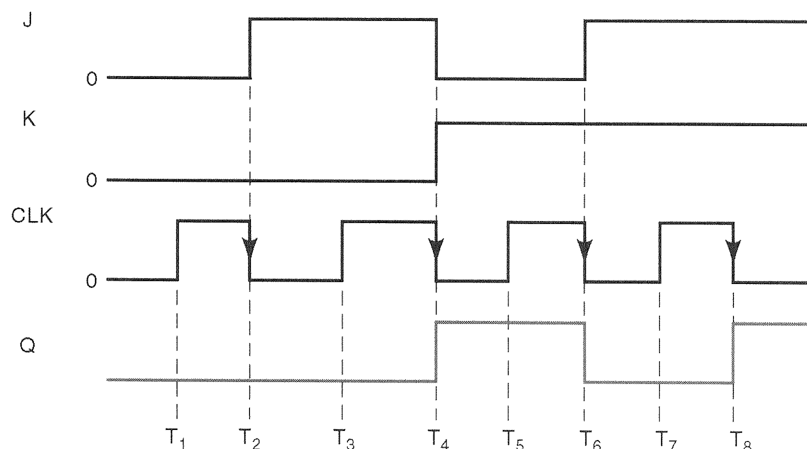


Fig. 5-38 Exemplo 5-11.

EXEMPLO 5-12

A Fig. 5-39(a) mostra uma situação em que o sinal de entrada *A* é gerado por uma chave sem trepidação que é acionada por um operador (um circuito para eliminar o efeito de trepidação da chave foi apresentado anteriormente no Exemplo 5-2). O sinal *A* vai para ALTO quando o operador aciona a chave e vai para BAIXO quando o operador libera a chave. Esta entrada *A* é usada para controlar a passagem do sinal de clock através da porta AND, de modo que pulsos de clock apareçam na saída *X* somente enquanto *A* for ALTO.

O problema com este circuito é que o sinal *A* é assíncrono e, portanto, pode mudar de estado em qualquer instante de tempo em relação ao sinal de clock, pois os momentos em que o operador atua ou libera a chave são essencialmente randômicos. Isto pode produzir pulsos de clock *parciais* na saída *X* se qualquer transição de *A* ocorrer enquanto o sinal de clock estiver em ALTO, conforme mostram as formas de onda da Fig. 5-39(b).

Esse tipo de saída normalmente é inaceitável, e portanto um método para prevenir o aparecimento de pulsos parciais em *X* deve ser desenvolvido. Uma solução é mostrada na Fig. 5-40(a). Descreva como este circuito resolve o problema e desenhe a forma de onda de *X* para a mesma situação da Fig. 5-39(b).

Solução

O sinal *A* é conectado à entrada *D* do FF *Q*, que por sua vez é disparado pela descida do sinal de clock. Assim, quando *A* vai para ALTO, *Q* só vai para ALTO na próxima descida do clock no instante T_1. Este nível ALTO em *Q* habilita a porta AND a passar os pulsos de clock *completos* subseqüentes para *X*, conforme mostra a Fig. 5-40(b).

Quando *A* retorna para BAIXO, *Q* só vai para BAIXO na próxima descida do clock em T_2. Logo, a porta AND só vai inibir os pulsos de clock após o pulso de clock que termina em T_2 passar através de *X*. Assim, a saída *X* contém somente pulsos de clock completos.

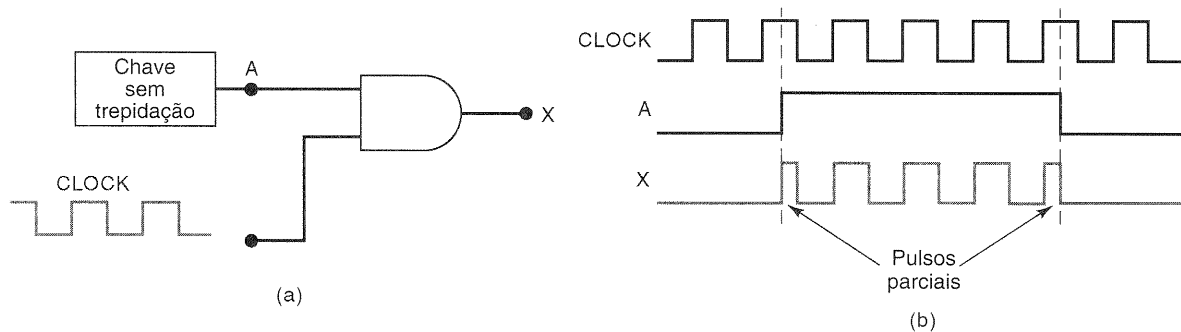


Fig. 5-39 O sinal assíncrono *A* pode produzir pulsos parciais em *X*.

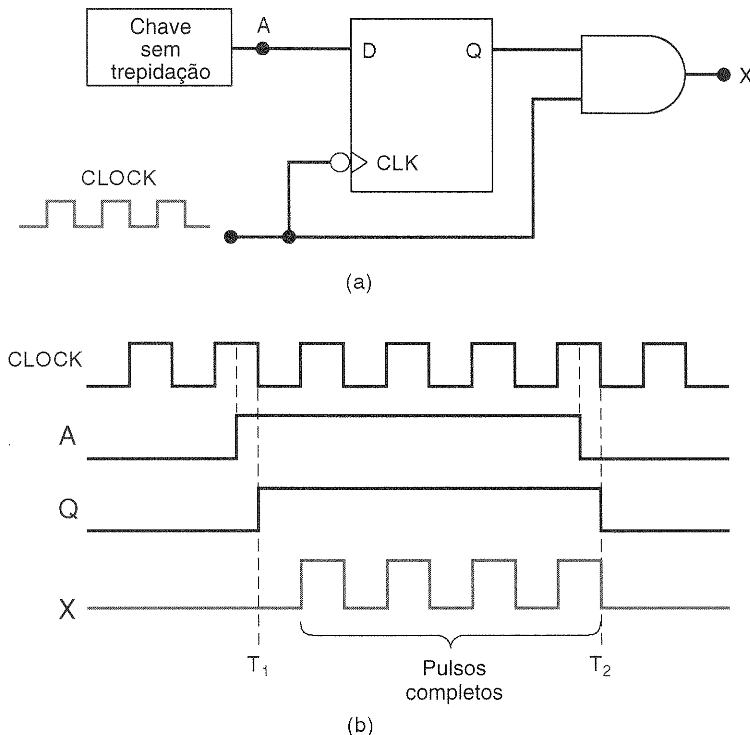


Fig. 5-40 Um flip-flop D disparado por transição é usado para sincronizar a habilitação da porta AND com a descida do clock.

5-16 DETECTANDO UMA SEQÜÊNCIA DE ENTRADA

Em diversas situações uma saída deve ser ativada somente quando as entradas foram ativadas em uma certa seqüência. Isto não pode ser realizado usando apenas a lógica combinacional, pois requer a capacidade de armazenamento dos FFs.

Por exemplo, uma porta AND pode ser usada para determinar quando duas entradas *A* e *B* estão ambas em ALTO, mas sua saída responderá do mesmo modo, não importando qual das entradas foi para ALTO primeiro. Mas suponha que desejamos gerar uma saída em ALTO *somente* se *A* for para ALTO e, então, um tempo depois, *B* for para ALTO. Um modo de realizar isto é mostrado na Fig. 5-41(a).

As formas de onda das Fig. 5-41(b) e (c) mostram que a saída *Q* vai para ALTO somente se *A* vai para ALTO antes que *B* vá para ALTO. Isto é porque *A* deve estar em ALTO de modo que a saída *Q* vá para ALTO na subida de *B*.

Para que este circuito funcione adequadamente, *A* deve ir para ALTO antes de *B*, pelo menos por um intervalo de tempo igual ao tempo de setup do FF.

5-17 ARMAZENAMENTO E TRANSFERÊNCIA DE DADOS

Com certeza, a utilização mais comum dos flip-flops é no armazenamento de dados ou informações. Os dados podem representar valores numéricos (como por exemplo números binários, números codificados em BCD), e são geralmente armazenados em grupos de flip-flops chamados **registradores**.

A operação mais freqüentemente executada com as informações armazenadas em um flip-flop ou em um registrador é a operação de **transferência de dados**. Isto envolve a transferência de informações entre um flip-flop (ou um registrador) e um outro. A Fig. 5-42 ilustra como uma transferência de dados pode ser feita usando flip-flops com clock dos tipos J-K, S-C ou D. Em cada caso, o valor lógico que está atualmente armazenado no FF *A* é transferido para o FF *B* na transição negativa do pulso TRANSFER. Portanto, após esta transição, a saída *B* terá o mesmo valor da saída *A*.

As operações de transferência vistas na Fig. 5-42 são exemplos de **transferência síncrona**, uma vez que ape-

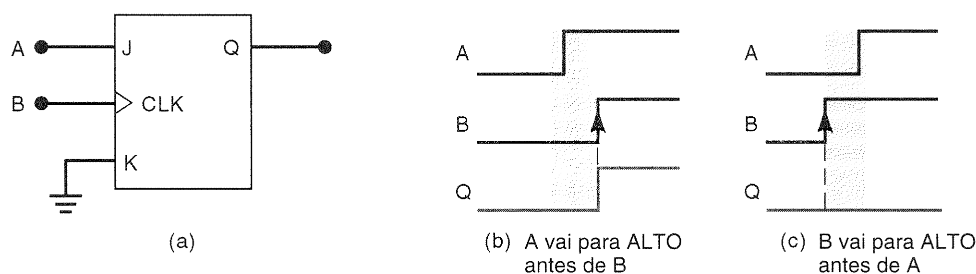


Fig. 5-41 Flip-flop J-K com clock usado para responder a uma determinada seqüência de entradas.

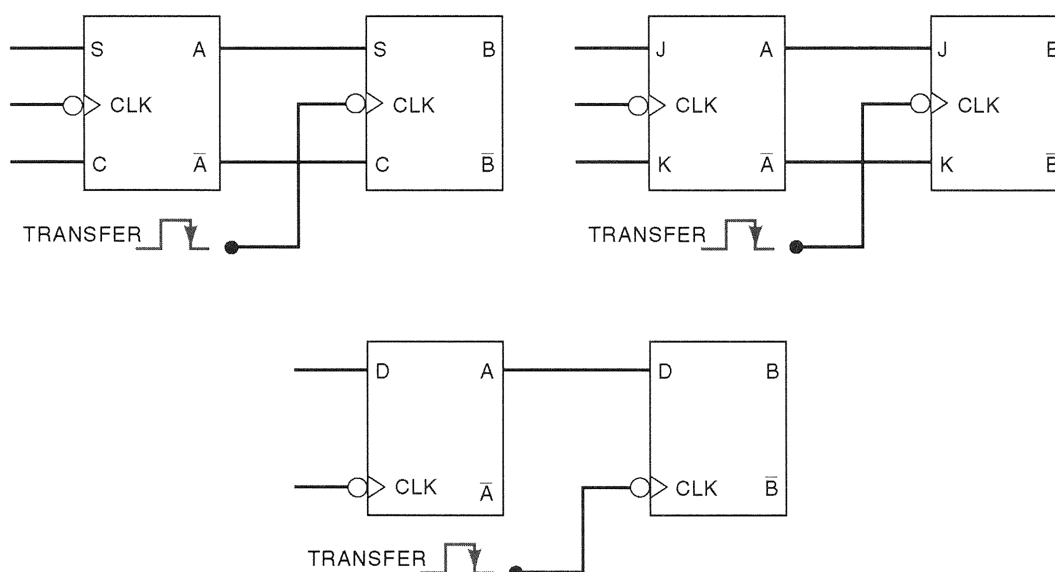


Fig. 5-42 Operação de transferência de dados síncrona em diversos tipos de FFs.

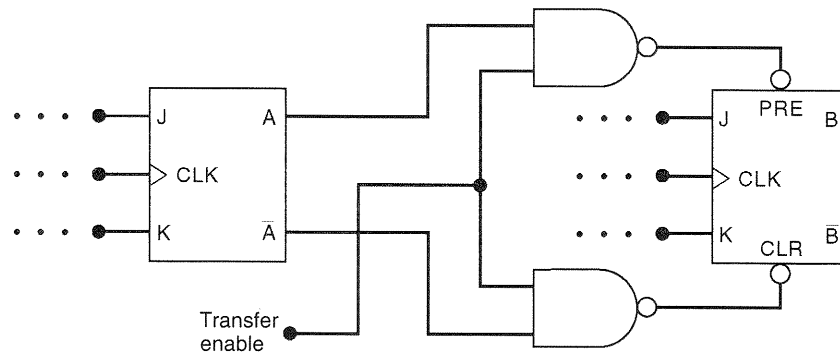


Fig. 5-43 Operação de transferência de dados assíncrona.

nas as entradas de controle síncronas e a entrada de clock foram utilizadas para realizar a transferência. Uma operação de transferência também pode ser realizada utilizando-se as entradas assíncronas de um flip-flop. A Fig. 5-43 mostra como uma **transferência assíncrona** pode ser feita utilizando-se as entradas PRESET e CLEAR de qualquer tipo de flip-flop. Neste caso, as entradas assíncronas são ativas em BAIXO. Quando a linha TRANSFER ENABLE é mantida em BAIXO, as saídas das portas NAND são mantidas em ALTO, não tendo nenhum efeito sobre a saída do flip-flop. Quando a linha TRANSFER ENABLE vai para ALTO, uma das saídas das portas NAND vai para nível BAIXO, dependendo do estado das saídas A e $\overline{A}$. Este nível BAIXO vai setar ou ressetar o FF B para o mesmo estado do FF A. Esta transferência assíncrona é feita independentemente das entradas síncronas e do clock. A transferência assíncrona é também chamada de **transferência por interferência,** porque o dado que está sendo transferido "interfere" no FF B, mesmo que suas entradas síncronas estejam ativas.

Transferência Paralela de Dados

A Fig. 5-44 mostra uma transferência de dados de um registrador para outro usando flip-flops do tipo D. O registrador X é composto dos FFs X_1, X_2 e X_3 e o registrador Y é composto dos FFs Y_1, Y_2 e Y_3. Quando o pulso de transferência é aplicado, o nível armazenado em X_1 é transferido para Y_1, o de X_2 para Y_2 e o de X_3 para Y_3. A transferência do conteúdo do registrador X para o registrador Y é síncrona. Ela também é chamada de **transferência paralela**, uma vez que o conteúdo de X_1, X_2 e X_3 foi transferido *simultaneamente* para Y_1, Y_2 e Y_3. Se uma **transferência serial** fosse realizada, o conteúdo do registrador X seria transferido, um bit de cada vez, para o registrador Y. Falaremos mais sobre este assunto na próxima seção.

É importante entender que a transferência paralela não altera o conteúdo do registrador que é a fonte das informações a serem transmitidas. Por exemplo, na Fig. 5-44, se os valores contidos nos registradores antes da ocorrência do pulso TRANSFER são $X_1X_2X_3 = 101$ e $Y_1Y_2Y_3 = 011$, após o

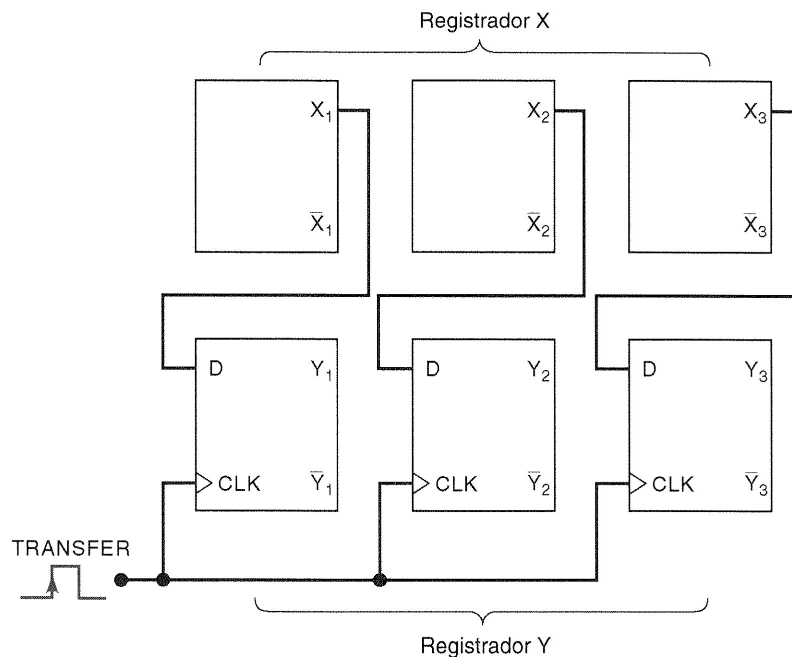


Fig. 5-44 Transferência paralela do conteúdo do registrador X para o registrador Y.

pulso TRANSFER, o conteúdo de ambos os registradores será 101.

Questões de Revisão

1. *Verdadeiro* ou *falso*: A transferência de dados assíncronos usa a entrada *CLK*.
2. Que tipo de flip-flop é o mais indicado para transferências síncronas porque necessita de um menor número de ligações entre um flip-flop e outro?
3. Se os flip-flops J-K fossem usados como os registradores na Fig. 5-44, quantas ligações seriam necessárias para conectar um registrador ao outro?
4. *Verdadeiro* ou *falso*: A transferência de dados síncrona necessita de menos circuitos que o modo assíncrono.

5-18 TRANSFERÊNCIA SERIAL DE DADOS: REGISTRADORES DE DESLOCAMENTO

Antes de descrevermos a operação de transferência serial de dados, devemos, primeiro, estudar a configuração básica de um *registrador de deslocamento*. Um **registrador de deslocamento** é um grupo de flip-flops interligados de tal forma que os números binários armazenados nos FFs são deslocados de um FF para o próximo, a cada pulso de clock. Você, sem dúvida, já viu registradores de deslocamento funcionando em dispositivos como uma calculadora eletrônica, onde os dígitos mostrados no display se deslocam toda vez que se fornece um novo número pelo teclado. Esta operação é similar à de um registrador de deslocamento.

A Fig. 5-45(a) mostra uma maneira de organizar flip-flops J-K para fazê-los funcionar como um registrador de deslocamento de 4 bits. Observe que os FFs estão conectados de tal modo que o valor da saída X_3 é transferido para X_2, o de X_2 é transferido para X_1 e o de X_1 para X_0. Isto significa que, quando ocorre uma transição negativa do pulso de deslocamento, cada FF assume o valor armazenado anteriormente pelo FF que está à sua esquerda. O flip-flop X_3 assume o valor determinado pelas condições presentes em suas entradas *J* e *K* quando a transição negativa ocorre. Por enquanto, vamos considerar que as entradas *J* e *K* são acionadas pelo sinal DATA IN, cuja forma de onda pode ser vista na Fig. 5-45(b). Também admitiremos que todos os flip-flops estão no estado 0 antes de os pulsos de deslocamento serem aplicados.

As formas de onda na Fig. 5-45(b) mostram como os dados de entrada são deslocados da esquerda para a direi-

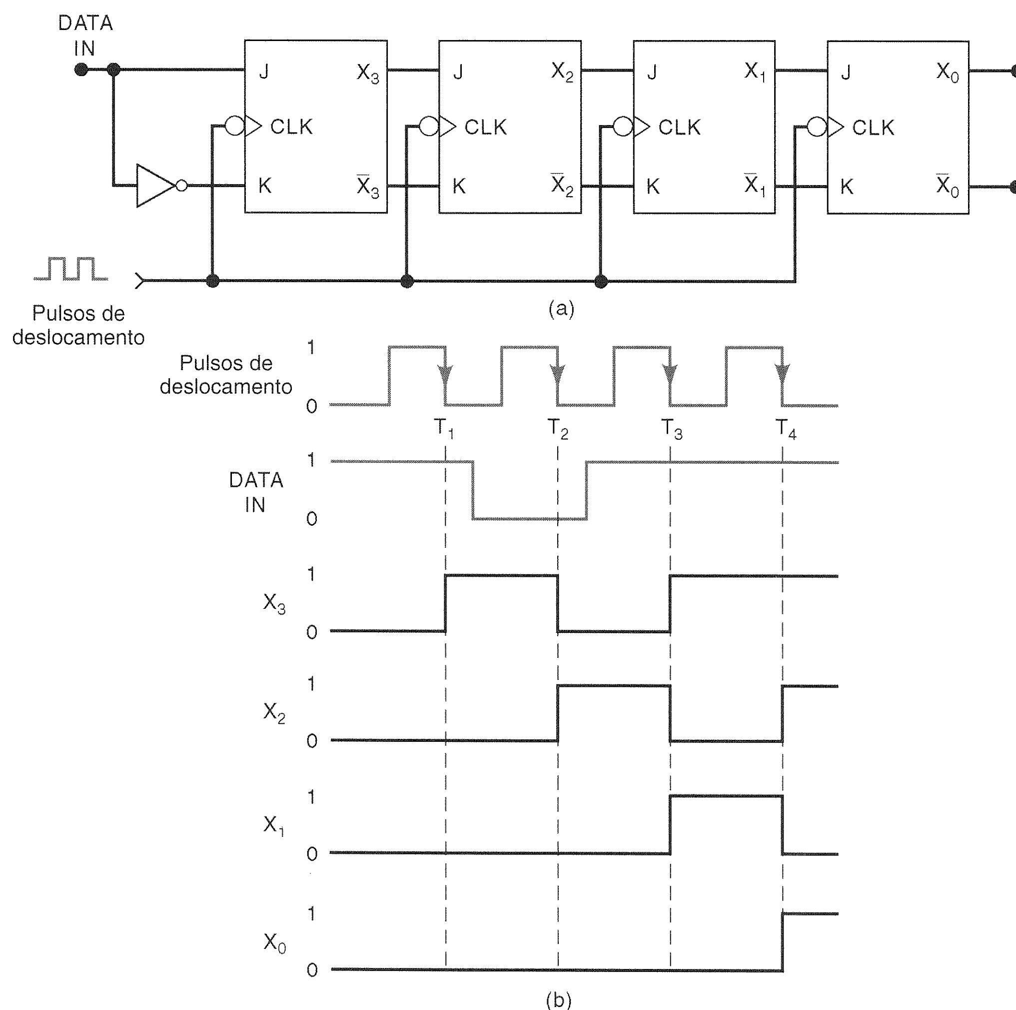


Fig. 5-45 Registrador de deslocamento de quatro bits.

ta, de um flip-flop para outro, à medida que os pulsos de deslocamento são aplicados. Quando a primeira transição negativa ocorre em T_1, cada um dos flip-flops X_2, X_1 e X_0 tem como condições de entrada $J = 0$ e $K = 1$, por causa do estado do flip-flop que está à sua esquerda. O flip-flop X_3 tem $J = 1$ e $K = 0$ devido ao valor de DATA IN. Então, em T_1, apenas X_3 vai para ALTO, enquanto todos os outros permanecem em BAIXO. Quando a segunda transição ocorre em T_2, o flip-flop X_3 tem $J = 0$ e $K = 1$ por causa de DATA IN. O flip-flop X_2 tem $J = 1$ e $K = 0$, por causa do nível ALTO presente em X_3. Os flip-flops X_1 e X_0 ainda têm $J = 0$ e $K = 1$. Então, em T_2 apenas o FF X_2 vai para ALTO, FF X_3 vai para BAIXO e os FFs X_1 e X_0 permanecem em BAIXO.

Um raciocínio semelhante pode ser usado para determinar como as formas de onda mudam em T_3 e em T_4. Observe que, em cada transição negativa dos pulsos de deslocamento, a saída de cada FF assume o nível que estava presente na saída do FF que está à sua esquerda, imediatamente *antes* da transição negativa. Obviamente, X_3 assume o valor que estava em DATA IN imediatamente antes da transição negativa.

Exigência Quanto ao Tempo de Hold

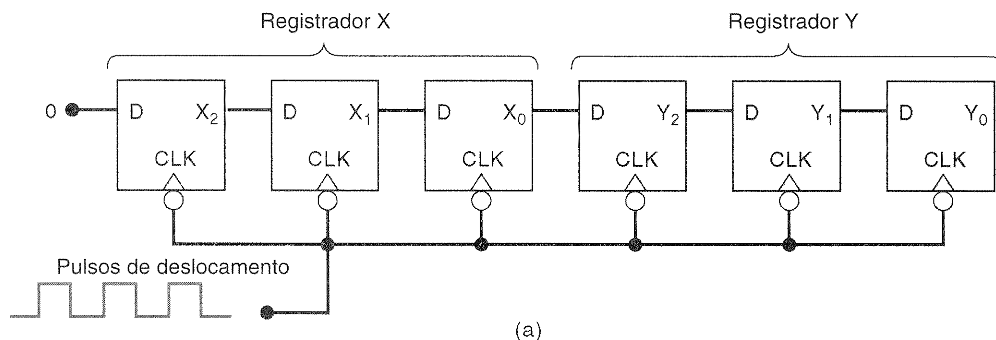
Neste arranjo de registrador de deslocamento, é necessário que os FFs tenham um tempo de hold muito pequeno, porque existirão momentos em que as entradas J e K estarão mudando de estado quase ao mesmo tempo que a transição na entrada *CLK*. Por exemplo, a saída X_3 muda de 1 para 0 em resposta à transição negativa em T_2, fazendo com que as entradas J e K, de X_2, mudem enquanto o nível na entrada *CLK* está mudando. Na verdade, devido ao atraso de propagação de X_3, as entradas J e K de X_2 não mudarão por um breve período após a transição negativa. Por esta razão, um registrador de deslocamento deve ser implementado usando FFs disparados por transição, cujo valor de t_H seja menor que o atraso de propagação (da saída em relação ao *CLK*). Este último requisito é tranqüilamente satisfeito pela maioria dos mais modernos FFs disparados por transição.

Transferência Serial entre Registradores

A Fig. 5-46(a) mostra dois registradores de deslocamento de três bits conectados de tal modo que o conteúdo do registrador X seja transferido serialmente (deslocado) para o registrador Y. Estamos usando flip-flops D para cada registrador porque este necessita de um menor número de ligações do que os flip-flops J-K. Observe como X_0, o último FF do registrador X, está conectado à entrada D de Y_2, o primeiro FF do registrador Y. Portanto, quando os pulsos de deslocamento são aplicados, a transferência de informação ocorre da seguinte maneira: $X_2 \rightarrow X_1 \rightarrow X_0 \rightarrow Y_2 \rightarrow Y_1 \rightarrow Y_0$. O flip-flop X_2 vai para o estado determinado por sua entrada D. Por enquanto, D será mantido em BAIXO, fazendo com que X_2 vá para BAIXO no primeiro pulso e depois permaneça neste estado.

Para ilustrar, vamos considerar que antes de os pulsos de deslocamento serem aplicados o conteúdo do registrador X seja 101 (isto é, $X_2 = 1$, $X_1 = 0$, $X_0 = 1$) e o do registrador Y, 000. Veja a tabela na Fig. 5-46(b), que mostra como os estados de cada flip-flop mudam à medida que os pulsos de deslocamento são aplicados. A partir desta tabela, podemos notar que:

1. Na transição negativa de cada pulso, cada FF assume o valor que foi armazenado no FF à sua esquerda, antes da ocorrência do pulso.
2. Após *três* pulsos, o 1 que estava inicialmente em X_2 está agora em Y_2, o 0 que estava inicialmente em X_1 está em



(a)

X_2	X_1	X_0	Y_2	Y_1	Y_0	
1	0	1	0	0	0	← Antes de os pulsos serem aplicados
0	1	0	1	0	0	← Depois do primeiro pulso
0	0	1	0	1	0	← Depois do segundo pulso
0	0	0	1	0	1	← Depois do terceiro pulso

(b)

Fig. 5-46 Transferência serial de informação do registrador X para o registrador Y.

Y_1 e o 1 que estava inicialmente em X_0 está agora em Y_0. Em outras palavras, o 101 armazenado no registrador X foi deslocado para o registrador Y, o conteúdo do registrador X é agora 000, e portanto ele perdeu seu valor inicial.
3. A transferência completa de *três* bits necessita de *três* pulsos de deslocamento.

A escolha de um tipo particular de transmissão (paralela ou serial) depende da aplicação e das especificações fornecidas. Geralmente, uma combinação dos dois tipos é utilizada para tirar proveito da *velocidade* da transferência paralela e da *economia e simplicidade* da transmissão serial. Adiante falaremos mais sobre transferência de informações.

EXEMPLO 5-13

Suponha os mesmos valores iniciais para os registradores X e Y da Fig. 5-46. O que acontece com o conteúdo de cada FF após a ocorrência do sexto pulso de deslocamento?

Solução

Se continuarmos o processo mostrado na Fig. 5-46(b) por mais três pulsos, vamos verificar que todos os FFs estarão em 0 após o sexto pulso. Uma outra maneira de se chegar a esta conclusão é a seguinte: o nível 0 constante na entrada D do flip-flop X_2 é deslocado a cada pulso, assim após seis pulsos os registradores estão preenchidos com 0s.

Questões de Revisão

1. *Verdadeiro ou falso*: O método mais rápido de transferir dados de um registrador para o outro é através da transferência paralela.
2. Qual é a maior vantagem da transferência serial sobre a paralela?
3. Observe a Fig. 5-46. Considere que os valores iniciais dos registradores são: $X_2 = 0$, $X_1 = 1$, $X_0 = 0$, $Y_2 = 1$, $Y_1 = 1$ e $Y_0 = 0$. Você também deve considerar que a entrada D de X_2 é mantida em ALTO. Determine o valor da saída de cada flip-flop após o quarto pulso de deslocamento.
4. Em que tipo de transferência de dados o emissor não perde os dados transferidos?

Operação de Deslocamento para a Esquerda

Os FFs na Fig. 5-46 podem ser facilmente conectados para que a informação seja deslocada da direita para a esquerda. Não existe nenhuma vantagem de fazer o deslocamento em um sentido em vez do outro. O sentido a ser escolhido pelo projetista dependerá da natureza da aplicação, como veremos a seguir.

Transferência Paralela *Versus* Serial

Na **transferência paralela**, todas as informações são transmitidas simultaneamente na ocorrência de um *único* pulso de transferência (Fig. 5-44), não importando o número de bits que estejam sendo transferidos. Na **transferência serial,** exemplificada na Fig. 5-46, a transferência completa de N bits de informação necessita de N pulsos de clock (três bits necessitam de três pulsos, quatro bits necessitam de quatro pulsos e assim por diante). Portanto, a transferência paralela é muito mais rápida do que a transferência serial utilizando registradores de deslocamento.

Na transferência paralela, a saída de cada FF no registrador X está conectada à entrada do FF correspondente no registrador Y. Na transferência serial, apenas o último FF no registrador X é conectado ao registrador Y. Portanto, de modo geral, a transferência paralela requer um maior número de interconexões entre o registrador emissor (X) e o receptor (Y) do que a transferência serial. Esta diferença torna-se mais significativa quando um grande número de bits de informação deve ser transferido. Esta consideração também é importante quando os registradores emissor e receptor estão distantes um do outro, pois isto determina quantas linhas de transmissão (fios) serão necessárias para a transmissão da informação.

5-19 DIVISÃO DE FREQÜÊNCIA E CONTAGEM

Observe a Fig. 5-47(a). Cada FF tem suas entradas J e K em nível 1, e portanto ele irá mudar de estado (comutar) sempre que o sinal em sua entrada CLK for de ALTO para BAIXO. Os pulsos de clock são aplicados apenas na entrada CLK do FF Q_0. A saída de Q_0 está conectada à entrada CLK de Q_1, e a saída de Q_1, por sua vez, está conectada à entrada CLK de Q_2. As formas de onda na Fig. 5-47(b) mostram como os FFs mudam de estado à medida que os pulsos são aplicados. Vamos destacar alguns pontos importantes:

1. O flip-flop Q_0 comuta na descida de cada pulso de clock. Portanto, a forma de onda da saída Q_0 tem uma freqüência que é exatamente igual à metade da freqüência do sinal de clock.
2. O flip-flop Q_1 comuta toda vez que a saída Q_0 vai de ALTO para BAIXO. A forma de onda de Q_1 tem freqüência igual à metade da freqüência da saída Q_0 e, portanto, um quarto da freqüência do sinal de clock.
3. O flip-flop Q_2 comuta cada vez que a saída Q_1 vai de ALTO para BAIXO, logo, a forma de onda de Q_2 tem freqüência igual à metade da freqüência de Q_1 e, portanto, um oitavo da freqüência do sinal de clock.
4. Cada forma de onda é uma onda quadrada (50% de taxa de ciclo*).

Conforme foi descrito anteriormente, cada FF divide a freqüência de entrada por 2. Portanto, se adicionássemos um quarto FF a esta cadeia, ele teria freqüência igual a um

*Taxa de ciclo é a razão entre o tempo em nível ALTO e o período da forma de onda. É comum a utilização do termo em inglês, *duty cycle*. (*N. do T.*)

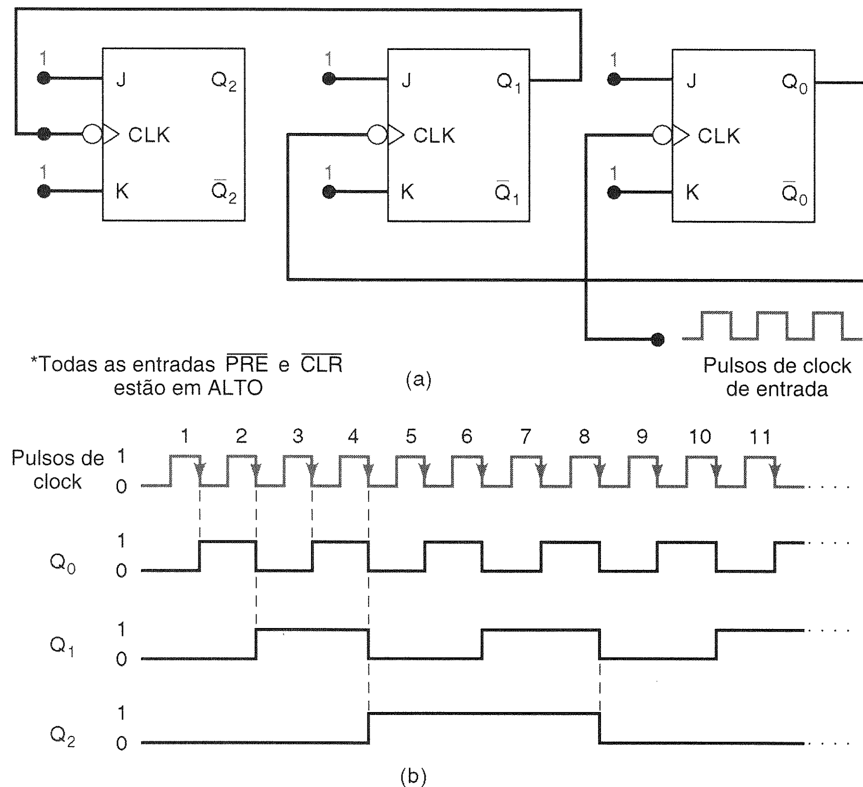


Fig. 5-47 Flip-flops J-K conectados para formar um contador de três bits (módulo 8).

dezesseis avos da freqüência do sinal de clock, e assim por diante. Usando o número apropriado de FFs, este circuito poderia dividir uma freqüência por qualquer potência de 2. Especificamente, utilizando *N* flip-flops produziríamos uma freqüência de saída no último flip-flop que seria igual a $1/2^N$ da freqüência de entrada.

Esta aplicação de flip-flops é chamada de **divisão de freqüência**. Muitas aplicações necessitam de divisão de freqüência. Por exemplo, seu relógio de pulso, que é sem dúvida um relógio "quartz". Falaremos mais sobre osciladores a cristal mais adiante neste capítulo, mas o termo *relógio quartz* significa que um cristal de quartzo é utilizado, em um oscilador, para gerar uma freqüência bastante estável. A freqüência natural de ressonância do cristal de quartzo do seu relógio é em torno de 1 MHz ou mais. Para que o mostrador dos segundos seja atualizado a cada 1 segundo, a freqüência do oscilador é *dividida* para gerar uma freqüência de saída bastante estável e precisa de 1 Hz.

Operação de Contagem

Além de funcionar como um divisor de freqüência, o circuito da Fig. 5-47 também funciona como um **contador binário**. Isto pode ser demonstrado observando-se a seqüência de estados dos FFs após a ocorrência de cada pulso de clock. A Fig. 5-48 apresenta os resultados em uma **tabela de estados**. Vamos dizer que $Q_2Q_1Q_0$ representam um número binário onde Q_2 está na posição 2^2, Q_1 está na posição 2^1 e Q_0 está na posição 2^0. Os primeiros oito estados de $Q_2Q_1Q_0$ devem ser reconhecidos como a seqüência de contagem binária de 000 a 111. Após a primeira transição negativa, os FFs estão no estado 001 ($Q_2 = 0$, $Q_1 = 0$ e $Q_0 = 1$), que representa 001_2 (equivalente ao decimal 1). Após a segunda transição negativa, os FFs estão no estado 010_2, que é equivalente a 2_{10}. Após três pulsos, eles estão em $011_2 = 3_{10}$; após quatro pulsos, eles estão em $100_2 = 4_{10}$ e assim sucessivamente, até que após 7 pulsos eles estão em $111_2 = 7_{10}$. Na oitava transição negativa, os FFs retornam

2^2 Q_2	2^1 Q_1	2^0 Q_0	
0	0	0	Antes dos pulsos de clock serem aplicados
0	0	1	Após pulso #1
0	1	0	Após pulso #2
0	1	1	Após pulso #3
1	0	0	Após pulso #4
1	0	1	Após pulso #5
1	1	0	Após pulso #6
1	1	1	Após pulso #7
0	0	0	Após pulso #8 retorna a 000
0	0	1	Após pulso #9
0	1	0	Após pulso #10
0	1	1	Após pulso #11
.	.	.	. . . .
.	.	.	. . . .
.	.	.	. . . .

Fig. 5-48 A tabela de estados dos flip-flops mostra a seqüência de contagem binária.

ao estado 000, e a seqüência binária se repete para os pulsos seguintes.

Então, para os primeiros sete pulsos de entrada, o circuito funciona como um contador binário, no qual o estado dos FFs representa o número binário equivalente ao número de pulsos que já ocorreram. Este contador pode contar até $111_2 = 7_{10}$ antes de retornar a 000.

Diagrama de Transição de Estados

Uma outra maneira de mostrar como os estados dos FFs mudam após cada pulso de clock é utilizar o **diagrama de transição de estados**, como pode ser visto na Fig. 5-49. Cada círculo representa um estado possível, como está indicado pelo número binário que está dentro do círculo. Por exemplo, o círculo que contém o número 100 representa o estado 100 (isto é, $Q_2 = 1$, $Q_1 = Q_0 = 0$).

As setas que ligam um círculo a outro mostram como um estado muda para outro quando o pulso de clock é aplicado. Apenas olhando para um estado em particular, podemos ver o estado que o precede e aquele que o sucede. Por exemplo, olhando para o estado 000, podemos ver que este estado é alcançado sempre que a contagem é 111 e um pulso de clock é aplicado. De modo semelhante, podemos ver que o estado 000 é sempre seguido pelo estado 001.

Usaremos os diagramas de transição de estados para nos ajudar a descrever, analisar e projetar contadores e outros circuitos seqüenciais.

Módulo do Contador

O contador da Fig. 5-47 possui $2^3 = 8$ estados diferentes (000 a 111). Dizemos que este é um contador de *módulo 8*, onde o valor do **módulo** indica o número de estados da seqüência binária. Se um quarto FF fosse adicionado, a seqüência de estados contaria, em binário, de 0000 a 1111, num total de 16 estados. Este seria um contador de *módulo 16*. De um modo geral, se *N* flip-flops estão conectados na configuração mostrada na Fig. 5-47, o contador resultante terá 2^N estados diferentes, e portanto será um contador de módulo 2^N. Ele será capaz de contar até $2^N - 1$ antes de retornar ao estado 0.

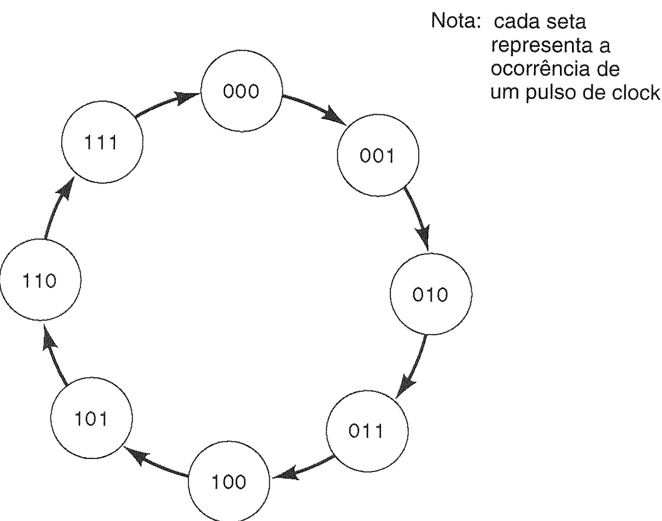


Fig. 5-49 O diagrama de transição de estados mostra como os estados do contador mudam a cada pulso de clock.

O módulo de um contador também indica a relação entre a freqüência de entrada e a freqüência de saída no último flip-flop. Por exemplo, um contador de quatro bits possui quatro FFs, onde cada um representa um dígito binário (bit) e é um contador de módulo 16. Ele pode contar até 15 $(2^4 - 1)$, e também pode ser usado para dividir a freqüência de entrada por 16 (o módulo do contador).

Estudamos apenas um contador binário elementar. Estudaremos contadores com muito mais detalhes no Capítulo 7.

EXEMPLO 5-14

Consideremos que o contador de módulo 8 da Fig. 5-47 está no estado 101. Qual será o estado (a contagem) após 13 pulsos terem sido aplicados?

Solução

Localize o estado 101 no diagrama de transição de estados. Siga o diagrama por 8 mudanças de estado. Você deve ter retornado ao estado 101. Agora continue por mais cinco mudanças de estado (fazendo um total de 13). Você deve estar agora no estado 010.

Observe que, como este é um contador de módulo 8, ele necessita de oito transições de estado para fazer uma excursão completa no diagrama e retornar ao estado inicial.

EXEMPLO 5-15

Considere um circuito contador que possui seis FFs conectados segundo o diagrama da Fig. 5-47 (isto é, Q_5, Q_4, Q_3, Q_2, Q_1, Q_0).

(a) Determine o módulo do contador.
(b) Determine a freqüência na saída do último FF (Q_5) quando a freqüência de entrada é 1 MHz.
(c) Qual é a faixa de contagem para este contador?
(d) Suponha que o estado (contagem) inicial é 000000. Qual será o estado deste contador após 129 pulsos?

Solução

(a) Módulo = $2^6 = 64$.
(b) A freqüência na saída do último FF será igual à freqüência de entrada do clock dividida pelo módulo do contador, isto é,

$$f(\text{em } Q_5) = \frac{1\text{ MHz}}{64} = 15{,}625\text{ kHz}$$

(c) O contador irá contar de 000000_2 até 111111_2 (0 a 63_{10}) num total de 64 estados. Observe que o número de estados é igual ao módulo do contador.
(d) Uma vez que este é um contador de módulo 64, a cada 64 pulsos de clock o contador é trazido ao seu estado

inicial. Portanto, após 128 pulsos de clock, a contagem retorna para 000000. O 129º pulso coloca o contador no estado 000001.

Questões de Revisão

1. Um sinal de clock de 20 kHz é aplicado a um flip-flop J-K com $J = K = 1$. Qual é a freqüência do sinal de saída do FF?
2. Quantos FFs são necessários para construir um contador que seja capaz de contar de 0 a 255_{10}?
3. O que é o módulo de um contador?
4. Qual é a saída do oitavo FF quando a freqüência de entrada do clock é de 512 kHz?
5. Se este contador começa em 00000000, qual será o seu estado após 520 pulsos?

5-20 APLICAÇÃO EM MICROCOMPUTADOR

Estamos apenas iniciando o nosso estudo de sistemas digitais e por isto você ainda não aprendeu muito sobre microprocessadores e microcomputadores. Entretanto, você pode entender como FFs são utilizados em uma aplicação controlada por microprocessador, sem se preocupar com todos os detalhes referentes à aplicação que você precisará saber mais tarde.

A Fig. 5-50 mostra uma unidade microprocessadora (MPU) onde suas saídas são usadas para transferir dados binários para o registrador *X*, que é constituído de 4 flip-flops do tipo D, X_3, X_2, X_1 e X_0. Um conjunto de saídas da MPU fornece o *endereço* e é formado pelas saídas A_{15}, A_{14}, A_{13}, A_{12}, A_{11}, A_{10}, A_9 e A_8. A maioria das MPUs possui pelo menos 16 linhas de endereço disponíveis, mas nem sempre todas elas são usadas. Um segundo conjunto de saídas da MPU é composto de quatro *linhas de dados* D_3, D_2, D_1 e D_0. A maioria das MPUs tem pelo menos oito linhas de dados disponíveis. O outro sinal da MPU é o sinal de clock, *CP*.

Lembre-se de que a MPU é a unidade central de processamento de um microcomputador, e sua função principal é executar um conjunto de instruções (programa) armazenado na memória do computador. Uma das instruções, que ele pode executar, é aquela que diz à MPU para transferir um número binário de um registrador de armazenamento interno da MPU para um outro registrador externo *X*. A execução desta instrução se dá nas seguintes etapas:

1. O endereço apropriado é colocado nas linhas de endereço (A_{15} a A_8) para selecionar o registrador *X* como destino dos dados a serem transferidos.
2. O número binário é colocado nas linhas de dados (D_3 a D_0).
3. Uma vez que as linhas de endereços e dados estão estáveis, a MPU gera o pulso de clock *CP* para o registrador, completando a transferência paralela de dados para *X*.

Existem muitas situações em que a MPU, sob o controle de um programa, transfere dados para registradores externos para controlar eventos externos. Por exemplo, as saídas dos FFs de um registrador podem ligar e desligar (através de circuitos de interface apropriados) dispositivos eletromecânicos como solenóides, relés, motores etc. Os dados que são enviados pela MPU para o registrador vão determinar que dispositivos devem estar ligados e quais devem estar desligados. Um outro exemplo bastante comum, é aquele que um registrador é utilizado para armazenar um número binário que servirá de entrada para um conversor digital-analógico (conversor D/A). A MPU transfere este número binário para o registrador, e o conversor gera uma tensão analógica que pode ser usada para controlar algo, como a posição de um feixe de elétrons em um tubo de imagem ou a velocidade de um motor.

EXEMPLO 5-16

(a) Que endereço deve ser gerado pela MPU para que os dados sejam transferidos para o registrador *X*?

(b) Considere que X_3-X_0 = 0110, A_{15}-A_8 = 11111111 e D_3-D_0 = 1011. Qual será o conteúdo de *X* após o pulso de *CP*?

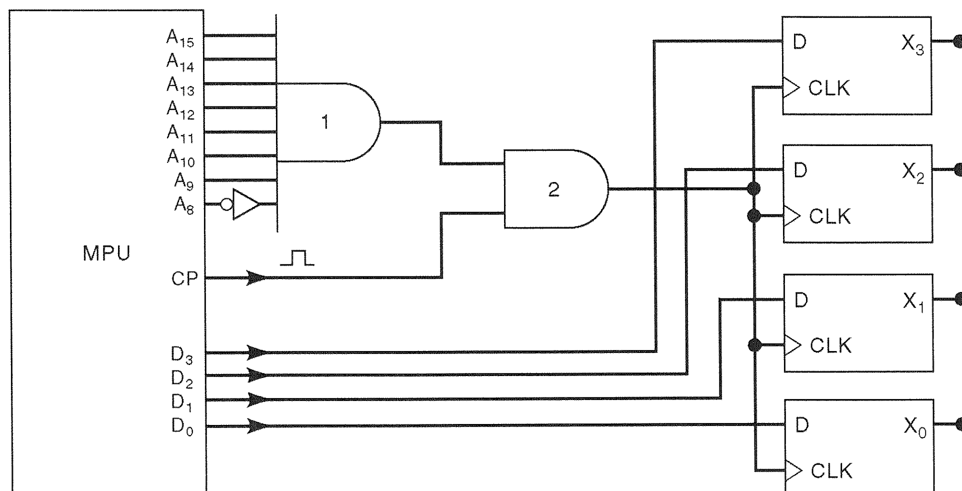


Fig. 5-50 Exemplo de um microprocessador transferindo dados binários para um registrador externo.

Solução

(a) Para que os dados sejam transferidos para *X*, o pulso de clock deve passar pela porta AND 2 para chegar às entradas *CLK* dos FFs. Isto acontecerá se a entrada superior da porta AND 2 estiver em ALTO. Isto significa que todas as entradas da porta AND 1 devem estar em ALTO, isto é, A_{15} até A_9 devem ser iguais a 1 e A_8 deve ser 0. Portanto, a presença do endereço 11111110 é necessária para permitir que os dados sejam transferidos para *X*.

(b) Com $A_8 = 1$, o nível BAIXO na entrada da porta AND 1 vai impedir que *CP* chegue à porta AND 2, e portanto os FFs não serão disparados e o conteúdo do registrador não será alterado.

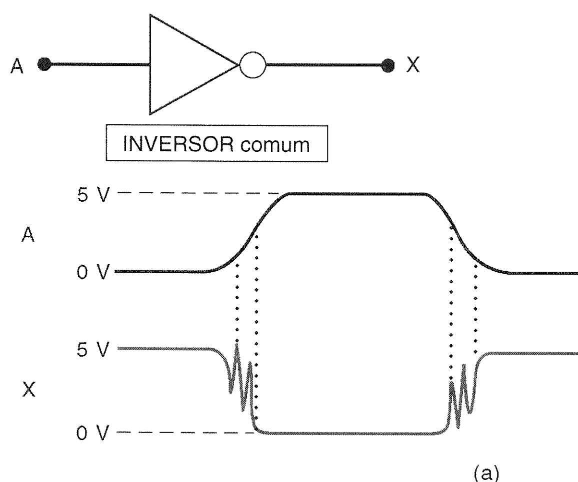
Questão de Revisão

1. Mostre como o 74HC175 da Fig. 5-34 pode ser usado como o registrador *X* da Fig. 5-50.

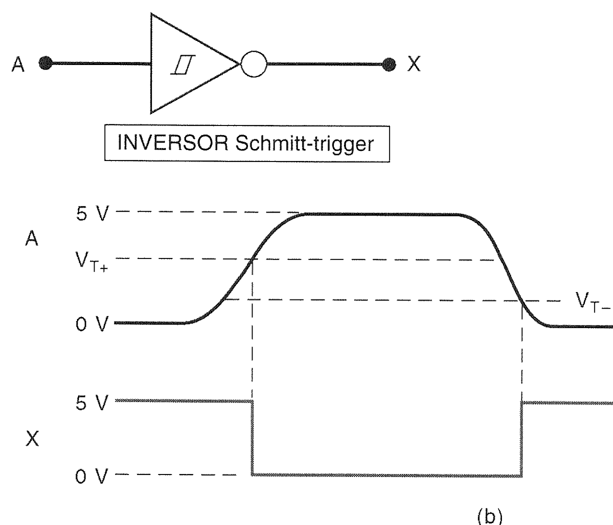
5-21 DISPOSITIVOS SCHMITT-TRIGGER

Um circuito **Schmitt-trigger** não é classificado como um flip-flop, mas possui um certo tipo de característica de memória que o torna bastante útil em determinadas situações. Uma destas situações está mostrada na Fig. 5-51(a). Neste caso, um INVERSOR comum está sendo acionado por uma entrada lógica que possui tempos de transição relativamente longos. Quando estes tempos de transição ultrapassam o valor máximo permitido (este depende de cada família lógica em particular), podem ocorrer oscilações nas saídas de portas lógicas e de INVERSORes à medida que o sinal de entrada passa pelo intervalo de indeterminação. Estas mesmas condições de entrada podem produzir disparos inesperados de flip-flops.

Um dispositivo que possui uma entrada do tipo Schmitt-trigger é projetado para aceitar sinais cuja transição é lenta e fornecer uma saída livre de oscilações. Esta saída geralmente possui tempos de transição muito rápidos (geralmente 10 ns) e é independente das características do sinal de entrada. A Fig. 5-51(b) mostra um INVERSOR Schmitt-trigger e sua resposta a uma entrada que varia lentamente.



Oscilações podem ocorrer na saída se os tempos de transição na entrada são muito longos



A saída possui transições rápidas, independentes dos tempos de transição da entrada

Fig. 5-51 (a) Se os tempos de transição são muito longos, a saída de um dispositivo lógico pode oscilar ou mudar de estado de modo imprevisível; (b) um dispositivo lógico com entradas do tipo Schmitt-trigger produzirá uma saída com transições rápidas.

Se você examinar as formas de onda da Fig. 5-51(b), deve notar que a saída não muda de ALTO para BAIXO até que a entrada ultrapasse a tensão de *limiar superior*, V_{T+}. Uma vez que a saída está em BAIXO, ela permanece neste estado, mesmo que a entrada caia abaixo de V_{T+} (esta é a característica de memória) e até que ela caia abaixo da tensão de *limiar inferior*, V_{T-}. Os valores destes dois limiares variam de uma família lógica para outra, mas V_{T-} sempre será menor que V_{T+}.

O INVERSOR Schmitt-trigger, e todos os outros dispositivos que possuem entradas deste tipo, utilizam um símbolo especial, mostrado na Fig. 5-51(b), para indicar que elas podem responder, de modo confiável, a entradas que variam lentamente. Os projetistas de circuitos lógicos usam CIs com entradas Schmitt-trigger para converter sinais que variam lentamente em sinais com transições rápidas e que podem acionar entradas de CIs comuns.

Vários CIs estão disponíveis com entradas Schmitt-trigger. O 7414, 74LS14 e o 74HC14 são CIs INVERSORes sêxtuplos com entradas Schmitt-trigger. O 74LS13 e o 74HC13 são CIs NANDs duplos de quatro entradas Schmitt-trigger.

Questões de Revisão

1. O que pode ocorrer quando um sinal com transições lentas é aplicado a um CI com entradas comuns?
2. Como um dispositivo lógico Schmitt-trigger opera de modo diferente ao de um dispositivo lógico comum?

5-22 MULTIVIBRADOR MONOESTÁVEL

Um circuito digital que está de algum modo relacionado com o flip-flop é o **monoestável** (abreviado MONO). Como o FF, o MONO possui duas saídas Q e $\overline{Q}$, que são o inverso uma da outra. Ao contrário do FF, o MONO possui apenas um estado de saída *estável* (normalmente $Q = 0$, $\overline{Q} = 1$), onde ele permanece até que seja disparado por um sinal de entrada. Uma vez disparado, o MONO muda para o estado oposto ($Q = 1$, $\overline{Q} = 0$). Ele permanece neste **estado quase-estável** por um período fixo de tempo t_p, que é geralmente definido pela constante de tempo *RC*, calculada a partir dos valores dos componentes externos R_T e C_T. Depois de um tempo t_p, os pulsos de saída OS retornam ao seu estado de repouso até serem disparados outra vez.

Existem dois tipos de MONOs disponíveis em circuitos integrados: os **MONOs redisparáveis*** e os **MONOs não-redisparáveis**.

Monoestável Não-redisparável

As formas de onda na Fig. 5-52(b) mostram a operação de um monoestável não-redisparável que é disparado na transição positiva da entrada *T*. Existem pontos importantes que devemos destacar:

1. As transições positivas nos pontos *a*, *b*, *c* e *e* vão disparar o MONO para o seu estado quase-estável por um intervalo t_p, após o qual ele automaticamente retorna ao estado estável.
2. As transições positivas nos pontos *d* e *f* não têm efeito no MONO porque este já se encontra no estado quase-estável. O MONO só pode ser disparado no seu estado estável.
3. A duração do pulso de saída é sempre a mesma, e independente da duração dos pulsos de entrada. Como afirmamos anteriormente, t_p depende apenas de R_T, C_T e do circuito interno do MONO. Um MONO típico possui um t_p que é dado pela expressão $t_p = 0,7\ R_TC_T$.

Monoestável Redisparável

O MONO redisparável funciona de modo similar ao MONO não-redisparável, mas com uma grande diferença: *ele pode ser disparado novamente durante seu estado quase-estável, iniciando um novo pulso de saída de duração t_p*. A Fig. 5-53(a) compara a resposta de ambos os tipos de monoestáveis com $t_p = 2$ ms. Vamos examinar estas formas de onda.

Ambos os tipos de MONO respondem ao primeiro pulso de disparo em $t = 1$ ms, indo para ALTO por 2 ms, e retornando depois a BAIXO. O segundo pulso de disparo, em $t = 5$ ms, dispara os dois monoestáveis que vão para ALTO. O terceiro pulso, em $t = 6$ ms, não tem efeito sobre o MONO não-redisparável, uma vez que ele já se encontra no seu estado quase-estável. Entretanto, este pulso vai *disparar* o MONO redisparável, que permanece em ALTO por 2 ms *após* este terceiro pulso.

Um MONO redisparável gera um pulso de saída de duração t_p toda vez que um pulso de disparo é aplicado, independentemente do estado atual de sua saída *Q*. Na verdade, o pulso de disparo pode ser aplicado em uma taxa alta o suficiente para que o MONO seja sempre redisparado antes do fim do intervalo t_p, fazendo com que a saída *Q* permaneça em ALTO. Isto é mostrado na Fig. 5-53(b), onde oito pulsos, com intervalo de 1 ms entre eles, são aplicados. A saída *Q* retorna para BAIXO somente 2 ms após o último pulso de disparo.

Dispositivos Reais

Vários CIs de monoestáveis estão disponíveis, tanto na versão redisparável quanto na versão não-redisparável. O 74121 é um CI com um único monoestável não-redisparável. O 74221, 74LS221 e o 74HC221 são CIs com dois monoestáveis não-redisparáveis. O 74122 e o 74LS122 são CIs com um único monoestável redisparável. O 74123, 74LS123 e o 74HC123 são CIs com dois monoestáveis redisparáveis.

A Fig. 5-54(a) mostra o símbolo tradicional para o 74121, CI com um monoestável não-redisparável. Observe que ele contém portas lógicas internas para permitir que as entradas A_1, A_2 e B possam disparar o MONO de várias maneiras. A entrada *B* é do tipo Schmitt-trigger e portanto permite que sinais com transições lentas disparem o MONO de modo confiável. Os pinos indicados por R_{INT}, R_{EXT}/C_{EXT} e C_{EXT} são usados para conectar o capacitor e o resistor externos, com valores escolhidos para obter a duração desejada para o pulso de saída. A Fig. 5-54(b) é o símbolo IEEE/ANSI para

*É bastante comum o uso do termo "retrigável" (oriundo da palavra inglesa *retriggerable*) como sinônimo de redisparável. (*N. do T.*)

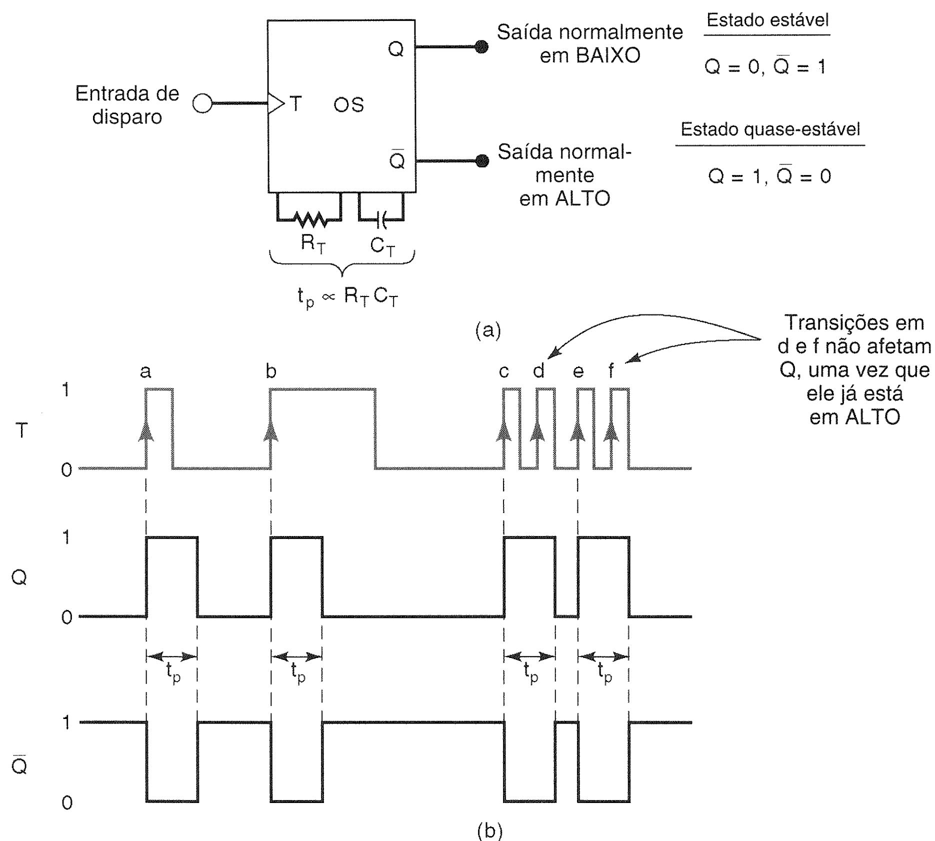


Fig. 5-52 Símbolo de um MONO e suas formas de onda típicas para a operação no modo não-redisparável.

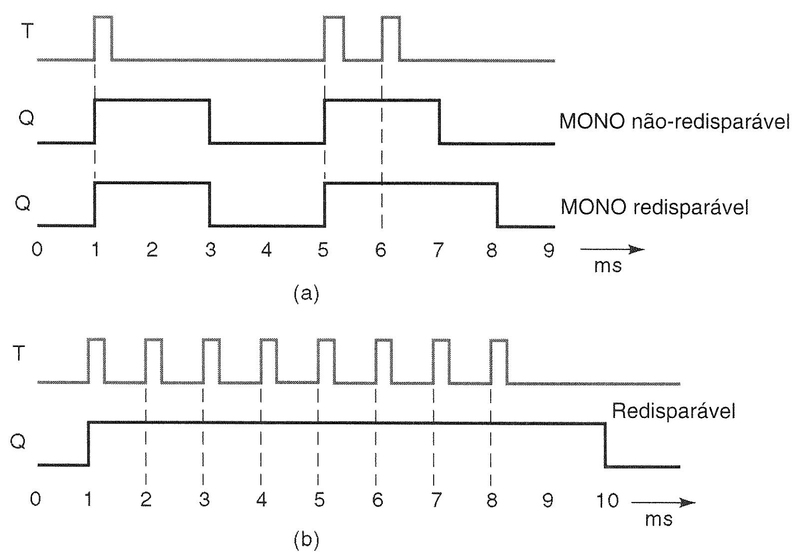


Fig. 5-53 (a) Comparação das respostas de MONOs redisparáveis e não-redisparáveis para t_p = 2 ms; (b) MONO redisparável inicia um novo intervalo t_p toda vez que recebe um pulso de disparo.

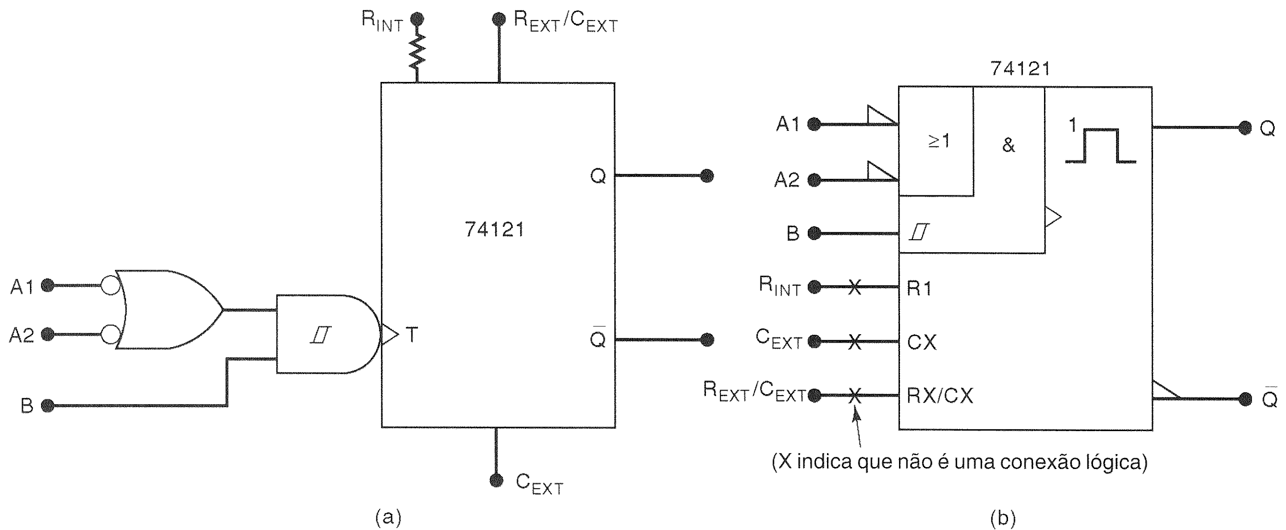


Fig. 5-54 Símbolos para o MONO não-redisparável 74121. (a) tradicional; (b) IEEE/ANSI.

o MONO não-redisparável 74121. Observe como este símbolo representa as portas lógicas. Observe também a presença de um pequeno pulso com o número 1 na frente. Isto indica que este dispositivo é um MONO não-redisparável. O símbolo IEEE/ANSI para um MONO redisparável nunca teria o número 1 na frente do pulso.

Multivibrador Monoestável

O monoestável tem este nome porque possui apenas um estado estável. Monoestáveis têm poucas aplicações na maioria dos sistemas síncronos, e projetistas experientes geralmente os evitam porque são susceptíveis a falsos disparos por ruídos espúrios. Eles são utilizados, geralmente, em aplicações simples de temporização. Vários exercícios no fim do capítulo mostram como um MONO é usado.

Questões de Revisão

1. Na ausência do pulso de disparo, qual será o estado da saída do MONO?
2. *Verdadeiro* ou *falso*: Quando um MONO não-redisparável se encontra no seu estado quase-estável, um pulso de disparo não afeta a saída.
3. O que determina o valor de t_p para um MONO?
4. Descreva como os MONOs redisparáveis funcionam de modo diferente dos não-redisparáveis.

5-23 ANÁLISE DE CIRCUITOS SEQÜENCIAIS

Muitos circuitos lógicos contêm FFs, MONOs e portas lógicas conectados para realizar uma operação específica. Freqüentemente, um sinal de clock principal é usado para fazer com que os níveis lógicos do circuito passem por uma seqüência particular de estados. De um modo geral, podemos analisar estes circuitos seqüenciais adotando o procedimento mostrado no exemplo a seguir.

EXEMPLO 5-17

Considere o circuito da Fig. 5-55. Inicialmente, todas as saídas dos FFs estão no estado 0, antes de os pulsos de clock serem aplicados. Estes pulsos são repetidos na freqüência de 1kHz. Determine as formas de onda de *X*, *Y*, *Z* e *W* por oito ciclos de clock.

Solução

Passo 1. Examine o circuito. Procure por configurações familiares, como contadores, registradores de deslocamento etc.

Os FFs *X*, *Y* e *Z* estão conectados como um contador de três bits que irá contar os pulsos de clock, desde que as entradas *J* e *K* do FF *Z*, que são acionadas pela saída da porta NAND, *W*, estejam em ALTO. As entradas da porta NAND são acionadas pelas saídas *X*, $\overline{Y}$ e $\overline{Z}$.

Passo 2. No diagrama do circuito, escreva os níveis lógicos presentes em cada saída e cada entrada antes da ocorrência do primeiro pulso de clock.

Os FFs estão inicialmente no estado 0. As entradas da porta NAND estão em 0, 1 e 1, respectivamente, e portanto *W* está em ALTO. Todas as entradas *J* e *K* estão em 1. Estes estados estão mostrados no diagrama do circuito.

Passo 3. Utilizando estas condições, determine os novos estados de cada FF em resposta ao primeiro pulso de clock.

Na descida do primeiro pulso de clock, *Z* vai comutar para o estado 1 e *X* e *Y* permanecem em BAIXO. Veja as formas de onda na Fig. 5-55.

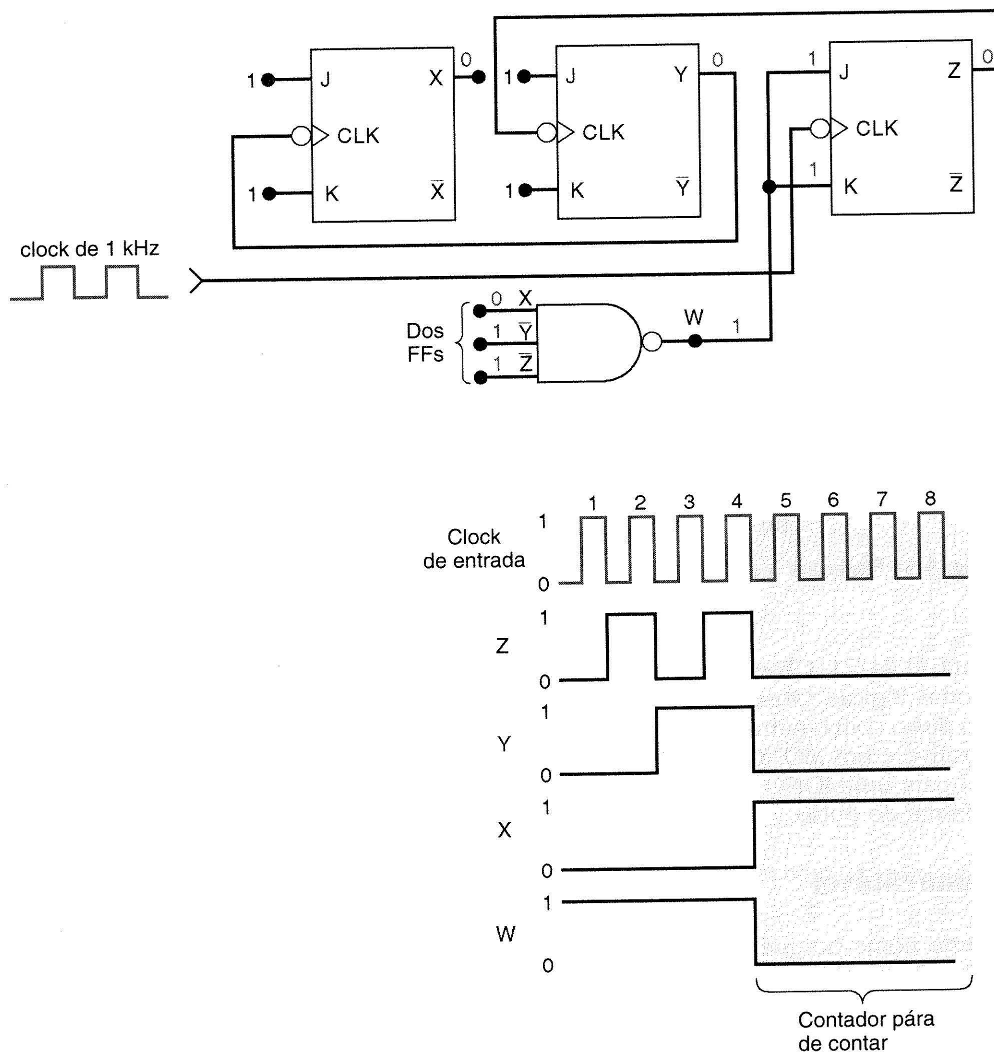


Fig. 5-55 Exemplo 5-17.

Passo 4. Volte e repita os passos 2 e 3 para o segundo pulso de clock, para o terceiro pulso e assim por diante.

Com Z agora em 1, as entradas da porta NAND estão em 0, 1 e 0, respectivamente, de modo que antes do segundo pulso de clock W ainda está em ALTO, todas as entradas J e K estão em ALTO e cada FF está pronto para comutar (você pode querer atualizar estes níveis no diagrama). A descida do segundo pulso de clock faz com que Z mude de 1 para 0. A transição negativa em Z, por sua vez, comuta Y de 0 para 1. X permanece em 0. Veja as formas de onda.

Antes do terceiro pulso de clock, as entradas da porta NAND são 0, 0 e 1, respectivamente, de modo que W ainda está em ALTO e todas as entradas J e K estão também em ALTO. O terceiro pulso de clock comuta Z de 0 para 1, enquanto X e Y permanecem inalterados. Veja as formas de onda.

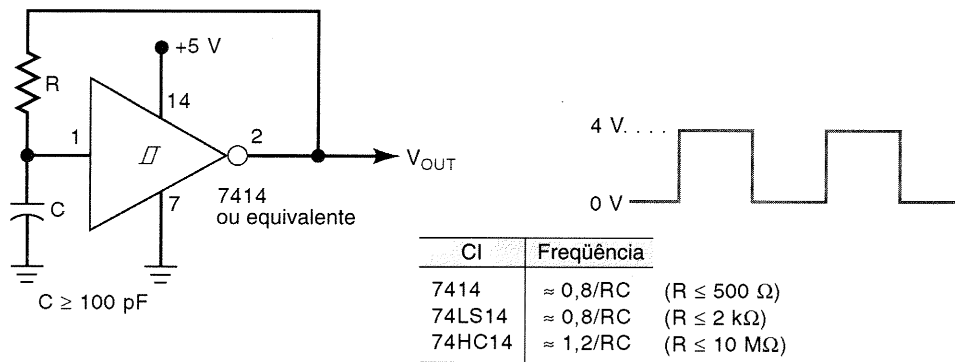
Antes do quarto pulso de clock, as entradas da porta NAND estão todas em 0, e portanto a saída W e todas as entradas J e K ainda estão em ALTO. O quarto pulso comuta Z de 1 para 0, o que, por sua vez, comuta Y de 1 para 0, fazendo com que X comute de 0 para 1. Veja as formas de onda.

Antes do quinto pulso de clock, as entradas da porta NAND estão todas em 1, de modo que a saída W está em BAIXO. Isto faz com que as entradas J e K do FF Z estejam em BAIXO, portanto sua saída não se altera. O quinto pulso de clock não vai ter qualquer efeito em Z, e nenhum dos níveis lógicos no circuito vai mudar. Na verdade, nenhuma das transições negativas a seguir causará mudança alguma, pois o contador está impedido de contar. Veja as formas de onda.

5-24 CIRCUITOS GERADORES DE CLOCK

Flip-flops possuem dois estados estáveis e, portanto, podem ser chamados de multivibradores biestáveis. Monoestáveis são chamados assim porque possuem apenas um estado estável. Um terceiro tipo é chamado de **multivibrador astável.** A saída deste tipo de circuito lógico oscila entre dois estados instáveis. Ele é bastante útil para gerar sinais de clock em sistemas digitais síncronos.

Vários tipos de multivibradores astáveis são de uso comum. Apresentaremos três deles, sem fazer qualquer tenta-



CI	Freqüência	
7414	≈ 0,8/RC	(R ≤ 500 Ω)
74LS14	≈ 0,8/RC	(R ≤ 2 kΩ)
74HC14	≈ 1,2/RC	(R ≤ 10 MΩ)

Fig. 5-56 Oscilador Schmitt-trigger usando um INVERSOR 7414. Um Schmitt-trigger NAND 7413 também pode ser usado.

tiva de analisar seu funcionamento. Eles são apresentados para que você possa construir um circuito gerador de clock, caso precise de um, para um projeto ou para testar circuitos no laboratório.

Oscilador Schmitt-Trigger

A Fig. 5-56 mostra como um INVERSOR Schmitt-trigger pode ser conectado como um oscilador. O sinal V_{OUT} é aproximadamente uma onda quadrada com uma freqüência que depende dos valores de R e C. A relação entre a freqüência e o valor da constante RC está mostrada na Fig. 5-56 para três tipos de INVERSORes Schmitt-trigger. Observe os limites máximos do valor de resistência para cada dispositivo. O circuito não funcionará se o valor da resistência não estiver abaixo destes limites.

Temporizador 555 Utilizado como um Multivibrador Astável

O **temporizador 555** é um dispositivo compatível com TTL e que pode funcionar de diversas maneiras. A Fig. 5-57 mostra como componentes externos podem ser conectados ao 555 para que ele funcione como um oscilador. Sua saída é uma forma de onda retangular repetitiva, que comuta entre dois níveis lógicos, e com o tempo de permanência em cada um destes níveis determinado pelos valores de R e C. As fórmulas para os tempos t_1 e t_2 e para o período total, T, podem ser vistas na figura. A freqüência do oscilador é, obviamente, igual ao inverso de T. As fórmulas apresentadas indicam que t_1 e t_2 não podem ser iguais, a não ser que R_A seja igual a zero. Isto não pode ser feito sem produzir um excesso de corrente através do dispositivo, o que significa que é impossível produzir uma onda quadrada perfeita com 50% de taxa de ciclo. É possível, entretanto, chegar bem próximo de 50% fazendo com que $R_B >> R_A$ (desde que você mantenha R_A maior do que 1 kΩ), conseguindo que $t_1 \approx t_2$.

EXEMPLO 5-18

Calcule a freqüência e a taxa de ciclo da saída do multivibrador astável 555 para $C = 0{,}001\ \mu F$, $R_A = 2{,}2\ k\Omega$ e $R_B = 100\ k\Omega$.

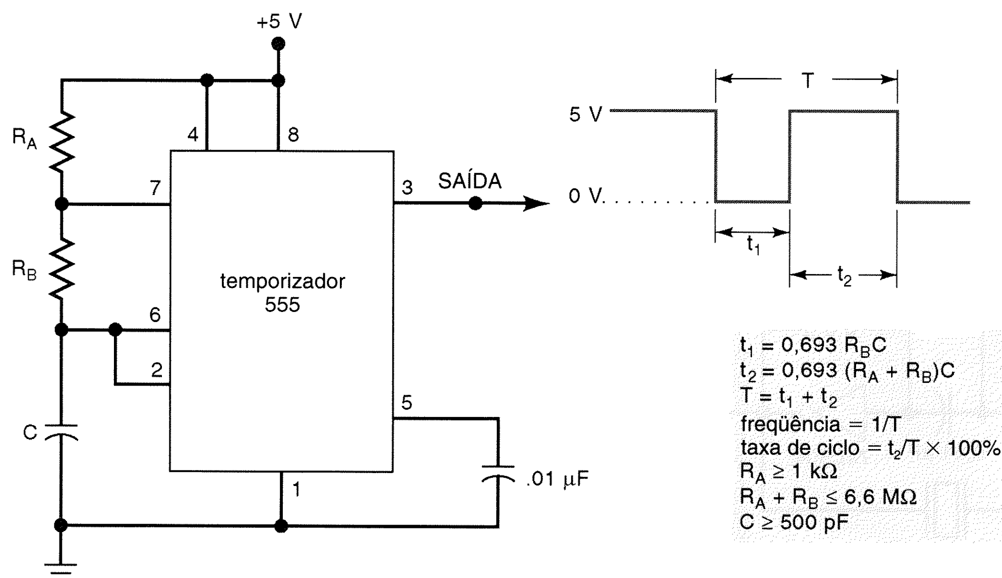


Fig. 5-57 Temporizador 555 usado como multivibrador astável.

Solução

$$t_1 = 0{,}693(100\ \text{k}\Omega)(0{,}001\ \mu\text{F}) = 69{,}3\ \mu\text{s}$$
$$t_2 = 0{,}693(102{,}2\ \text{k}\Omega)(0{,}001\ \mu\text{F}) = 70{,}7\ \mu\text{s}$$
$$T = 69{,}3 + 70{,}7 = 140\ \mu\text{s}$$
$$f = 1/140\ \mu\text{s}) = 7{,}29\ \text{kHz}$$
$$\text{taxa de ciclo} = 70{,}7/140 = 50{,}5\%$$

Observe que a taxa de ciclo é bem próxima de 50% (onda quadrada) porque R_B é muito maior que R_A. Podemos fazê-la ainda mais próxima de 50% fazendo R_B ainda maior que R_A. Por exemplo, você deve verificar que, se mudarmos o valor de R_A para 1 kΩ (valor mínimo permitido), teremos $f = 7{,}18$ kHz e a taxa de ciclo = 50,3%.

Geradores de Clock a Cristal

As freqüências de saída dos sinais provenientes dos circuitos descritos anteriormente dependem dos valores de resistores e capacitores, e portanto não são extremamente precisos ou estáveis. Mesmo que resistores variáveis sejam utilizados para que a freqüência desejada seja obtida através do ajuste destes resistores, os valores de *R* e *C* podem sofrer alterações devido a mudanças na temperatura ambiente e ao envelhecimento dos componentes, causando um desvio no valor da freqüência ajustada. Se a precisão e a estabilidade da freqüência são críticas, uma outra maneira de gerar sinais de clock pode ser usada: um **gerador de clock a cristal**. Ele utiliza um componente bastante preciso e estável chamado *cristal de quartzo*. Um pedaço de cristal de quartzo pode ser cortado, com forma e tamanho específicos, para vibrar (ressoar) em uma freqüência precisa e extremamente estável com a temperatura e com o envelhecimento. Cristais com freqüências de 10 kHz a 80 MHz estão prontamente disponíveis. Quando um cristal é colocado em determinados circuitos, estes podem oscilar em uma freqüência precisa e estável, igual à freqüência de ressonância do cristal. Dois destes circuitos são mostrados na Fig. 5-58.

O circuito da Fig. 5-58(a) é construído usando inversores TTL 74LS04. Poderíamos também usar o inversor Schmitt-trigger 74LS14. O valor de *R* está geralmente entre 300 e 1500 Ω, e depende do tipo do cristal utilizado e da sua freqüência. Este circuito é capaz de produzir freqüências de clock de até 20 MHz. O circuito da Fig. 5-58(b) usa inversores CMOS do CI 74HC04. Um valor típico para *R* é 100 kΩ. Este circuito é capaz de oscilar em freqüências de até 10 MHz.

Geradores de clock a cristal, como aqueles mostrados na Fig. 5-58, são usados em todos os sistemas baseados em microprocessadores e em microcomputadores, e também em qualquer aplicação na qual o sinal de clock necessite ser gerado com precisão. Veremos algumas destas aplicações nos próximos capítulos.

Questões de Revisão

1. Determine a freqüência aproximada do oscilador Schmitt-trigger que utiliza um 74HC14 com R = 10 kΩ e C = 0,005 μF.
2. Determine a freqüência aproximada e a taxa de ciclo de um oscilador 555 para $R_A = R_B = 2{,}2$ kΩ e $C =$ 2.000 pF.
3. Qual é a vantagem dos geradores de clock a cristal sobre aqueles baseados em *RC*?

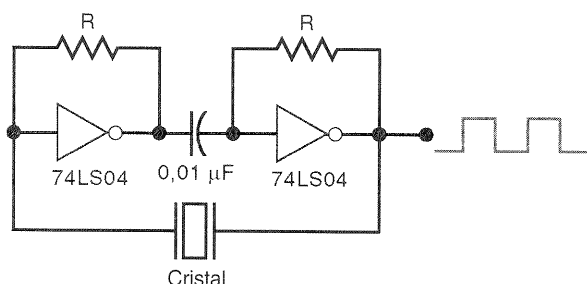
5-25 DEPURAÇÃO DE CIRCUITOS COM FLIP-FLOPS

Os CIs de flip-flops são susceptíveis aos mesmos tipos de falhas internas e externas que ocorrem em circuitos lógicos combinacionais. Todas as técnicas de depuração, discutidas no Cap. 4, podem ser prontamente aplicadas aos circuitos que contêm FFs tanto quanto àqueles com portas lógicas.

Por causa de suas características de memória, os circuitos com FFs com uma ou mais falhas freqüentemente exibem sintomas que não ocorreriam em circuitos combinacionais. Alguns deles são descritos a seguir.

Entradas em Aberto

Entradas desconectadas ou em flutuação de qualquer circuito lógico são especialmente susceptíveis para captar flu-



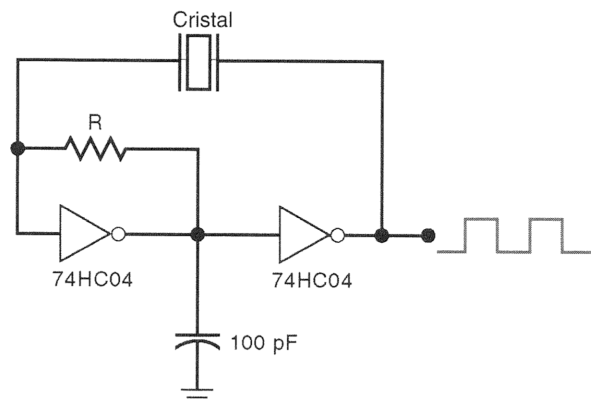


Fig. 5-58 Circuitos geradores de clock a cristal: (a) usando inversores TTL; (b) usando inversores CMOS.

tuações de tensão espúrias denominadas *ruído*. Se o ruído é grande o suficiente em amplitude e duração, a saída do circuito lógico pode alterar seu estado em resposta ao ruído. Em uma porta lógica, a saída retorna ao seu estado original quando o ruído termina. Num FF, entretanto, a saída permanece no seu novo estado por causa da sua característica de memória. Assim, o efeito de captação do ruído em qualquer entrada aberta é usualmente mais crítico para um FF ou latch do que para uma porta lógica.

As entradas mais susceptíveis do FF são aquelas que podem colocá-lo em um estado diferente, tais como: *CLK*, PRESET e CLEAR. Sempre que você vir uma saída de FF mudando de estado de modo errático, você pode considerar a possibilidade de uma conexão aberta em uma destas entradas.

EXEMPLO 5-19

A Fig. 5-59 mostra um registrador de deslocamento de três bits construído com flip-flops TTL. Inicialmente, todos os FFs estão no estado BAIXO antes de os pulsos de clock serem aplicados. Conforme os pulsos de clock são aplicados, cada transição positiva causa o deslocamento da informação de cada FF para o outro à sua direita. O diagrama mostra a seqüência "esperada" de estados dos FFs depois de cada pulso de clock. Já que $J_2 = 1$ e $K_2 = 0$, o flip-flop X_2 vai para ALTO no pulso de clock 1, e permanecerá lá durante os pulsos subseqüentes. Este nível ALTO se desloca para X_1 e depois para X_0 nos pulsos de clock 2 e 3, respectivamente. Assim, após o terceiro pulso, todos os FFs estarão em ALTO, e devem permanecer neste nível conforme os pulsos vão sendo aplicados.

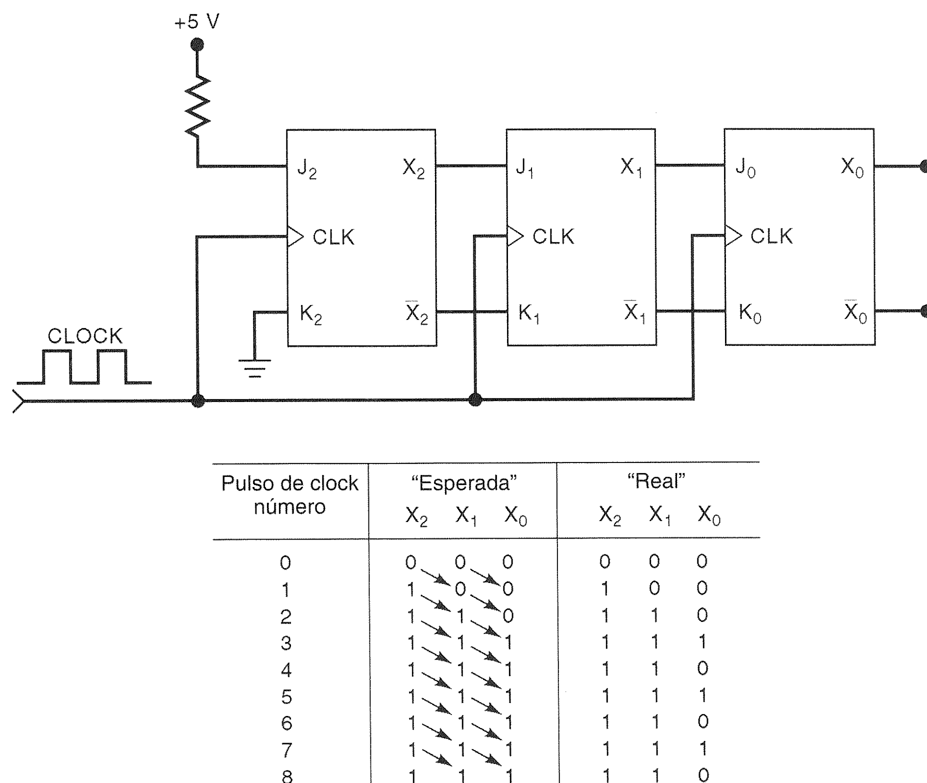
Agora vamos supor que a resposta "real" dos estados dos FFs é aquela mostrada no diagrama. Aqui os FFs mudam como esperado nos primeiros três pulsos de clock. A partir daí, o flip-flop X_0, em vez de ficar em ALTO, alterna entre ALTO e BAIXO. Que possível falha de circuito pode produzir este comportamento?

Solução

No segundo pulso, X_1 vai para ALTO. Isto deveria fazer $J_0 = 1$ e $K_0 = 0$, de modo que todos os pulsos de clock seguintes deveriam setar $X_0 = 1$. Em vez disso, vemos X_0 mudando de estado (comutando) em todos os pulsos depois do segundo. Esta comutação ocorreria se J_0 e K_0 estivessem ambos em ALTO. A falha mais provável é uma interrupção da ligação entre $\overline{X}_1$ e K_0. Lembre-se de que um dispositivo TTL responde a uma entrada aberta como se estivesse em nível lógico ALTO, portanto K_0 em aberto seria o mesmo que em ALTO.

Saídas em Curto

O exemplo a seguir ilustra como uma falha em um circuito com FF pode causar um sintoma enganador, o que resulta em mais tempo gasto para isolar a falha.



Pulso de clock número	"Esperada" X_2	X_1	X_0	"Real" X_2	X_1	X_0
0	0	0	0	0	0	0
1	1	0	0	1	0	0
2	1	1	0	1	1	0
3	1	1	1	1	1	1
4	1	1	1	1	1	0
5	1	1	1	1	1	1
6	1	1	1	1	1	0
7	1	1	1	1	1	1
8	1	1	1	1	1	0

Fig. 5-59 Exemplo 5-19.

EXEMPLO 5-20

Considere o circuito da Fig. 5-60 e examine as indicações da ponta de prova relacionadas na tabela. Existe um nível BAIXO na entrada *D* do FF quando pulsos são aplicados em sua entrada *CLK*, mas a saída *Q* falha em ir para BAIXO. O estudante que testa o circuito considera cada uma das possíveis falhas:

1. Z2-5 está internamente em curto com V_{CC}.
2. Z1-4 está internamente em curto com V_{CC}.
3. Z2-5 ou Z1-4 está externamente em curto com V_{CC}.
4. Z2-4 está internamente ou externamente em curto com a TERRA. Isto deveria manter $\overline{PRE}$ ativado e estaria se sobrepondo à entrada *CLK*.
5. Existe uma falha interna em Z2 que inibe *Q* de responder corretamente a suas entradas.

Depois de fazer as verificações necessárias com o ohmímetro, o estudante descarta as primeiras quatro possibilidades. Ele também verifica os pinos de V_{cc} e TERRA e constata que estão com valores de tensão adequados. Ele fica relutante em dessoldar Z2 do circuito até ter certeza de que o chip está ruim, e portanto decide examinar o sinal de clock. Ele utiliza um osciloscópio para visualizar a amplitude, a freqüência, as larguras de pulso e os tempos de transição. Constata que tudo está dentro das especificações do 74LS74. Finalmente, conclui que Z2 está danificado.

Ele retira o chip 74LS74 e troca-o por outro. Para seu espanto, o circuito com o novo chip se comporta exatamente do mesmo modo. Após coçar a cabeça, decide trocar o chip das portas NAND, embora sem saber por quê. Como era de se esperar, não há mudança na operação do circuito.

Tornando-se mais perplexo, ele recorda que seu professor de laboratório enfatizava a importância de se realizar uma verificação visual cuidadosa na placa de circuito impresso, e então começa a examiná-la com atenção. Ele detectou uma ponte de solda entre os pinos 6 e 7 de Z2. Removeu-a e testou o circuito que funcionou corretamente. Explique por que este erro produziu a operação observada.

Solução

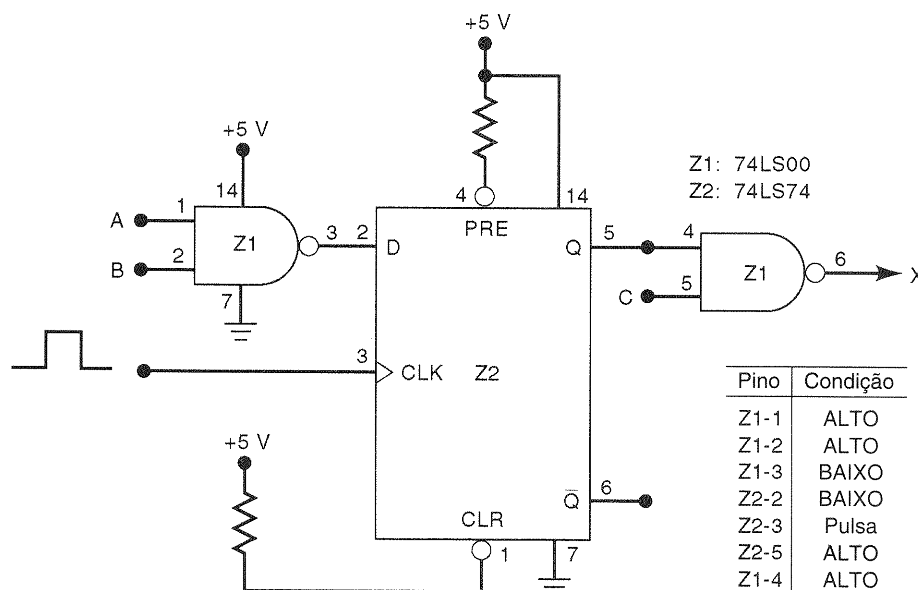
A ponte de solda estava colocando a saída $\overline{Q}$ em curto com a TERRA. Isto significa que a saída $\overline{Q}$ estava permanentemente fixa em nível BAIXO. Lembre-se de que, em todos os tipos de latch e FFs, as saídas $\overline{Q}$ e *Q* são internamente interligadas, de modo que o nível de uma afeta o nível da outra. Por exemplo, veja novamente o circuito interno de um flip-flop J-K na Fig. 5-23. Repare que um nível BAIXO constante em $\overline{Q}$ manteria uma das entradas da porta NAND 3 em BAIXO, e portanto *Q* teria que ficar em ALTO independentemente das condições para *J*, *K* e *CLK*.

O estudante aprendeu uma importante lição sobre depuração de circuitos com FFs. Ele aprendeu que ambas as saídas devem ser verificadas quanto a falhas, mesmo aquelas que não estejam conectadas a outros dispositivos.

Desalinhamento do Clock

Um dos problemas mais comuns em circuitos síncronos é o **desalinhamento do clock.** Um tipo de desalinhamento do clock ocorre quando um sinal de clock, devido aos atrasos de propagação, alcança as entradas *CLK* dos diversos FFs em instantes de tempo diferentes. Em várias situações o desalinhamento pode fazer com que o FF vá para um estado errado. Isto é mais bem ilustrado com um exemplo.

Observe a Fig. 5-61(a), onde o sinal *CLOCK1* está conectado diretamente ao FF Q_1 e indiretamente a Q_2 através da porta NAND e do INVERSOR. Ambos os FFs supostamente seriam disparados pela descida de *CLOCK1*, pois *X* está em ALTO. Se considerarmos que, inicialmente, $Q_1 = Q_2 = 0$ e $X = 1$, a transição negativa de *CLOCK1* deveria setar $Q_1 =$



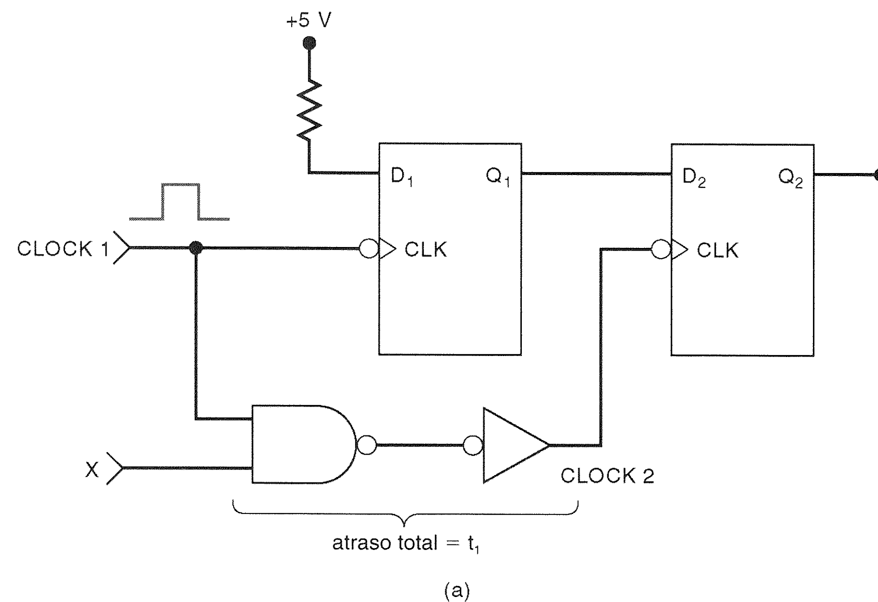
Pino	Condição
Z1-1	ALTO
Z1-2	ALTO
Z1-3	BAIXO
Z2-2	BAIXO
Z2-3	Pulsa
Z2-5	ALTO
Z1-4	ALTO

Fig. 5-60 Exemplo 5-20.

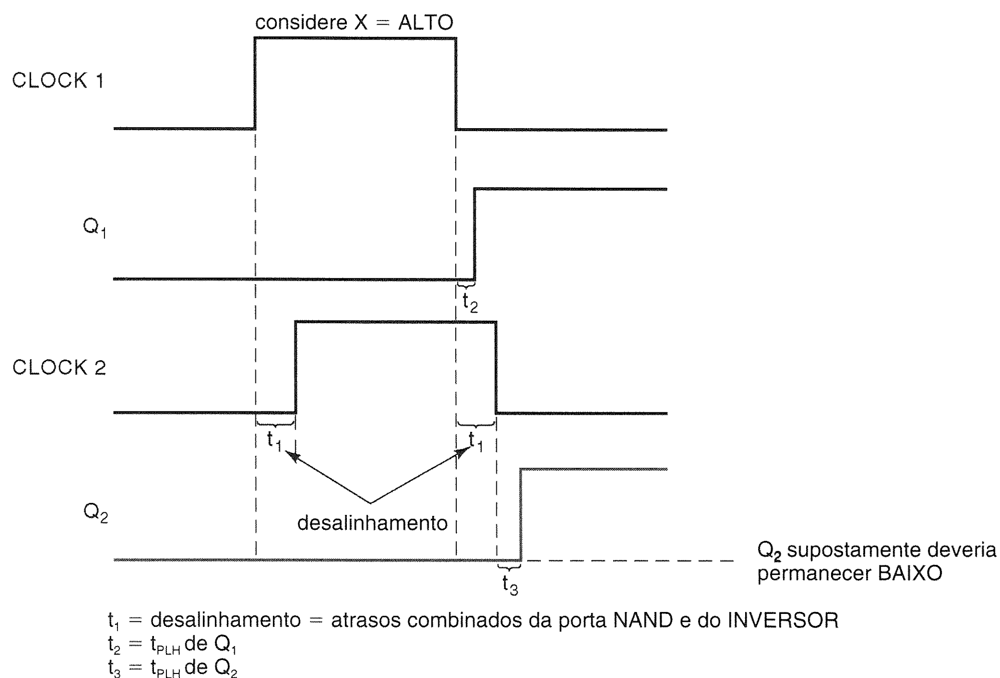
1 e não ter efeito sobre Q_2. As formas de onda na Fig. 5-61(b) mostram como o desalinhamento do clock pode produzir o disparo incorreto de Q_2.

Por causa dos atrasos de propagação combinados da porta NAND e do INVERSOR, as transições do sinal *CLOCK2* são atrasadas em relação ao *CLOCK1* por uma parcela de tempo t_1. A transição negativa de *CLOCK2* chega na entrada *CLK* de Q_2, após um tempo t_1 da transição negativa de *CLOCK1* aparecer na entrada *CLK* de Q_1. Esta parcela de tempo, t_1, é o desalinhamento do clock. A descida de *CLOCK1* faz Q_1 ir para ALTO depois de um tempo t_2, que é igual ao atraso de propagação t_{PLH} de Q_1. Se t_2 fosse menor do que o desalinhamento t_1, Q_1 estaria em ALTO na descida de *CLOCK2*, e isto poderia setar incorretamente $Q_2 = 1$ caso seu tempo de setup, t_S, fosse atendido.

Por exemplo, considere que o desalinhamento do clock é 40 ns e o t_{PLH} de Q_1 é 25 ns. Assim, Q_1 vai para ALTO 15 ns antes da descida de *CLOCK2*. Se o tempo de setup de Q_2 for menor do que 15 ns, Q_2 vai responder ao nível ALTO em sua entrada *D* na descida de *CLOCK2*, e Q_2 vai para



(a)



(b)

Fig. 5-61 O desalinhamento do clock ocorre quando dois flip-flops que supostamente são disparados simultaneamente são disparados em instantes de tempo ligeiramente diferentes devido a um atraso na chegada do sinal de clock do segundo flip-flop.

ALTO. Isto, naturalmente, não é a resposta esperada para Q_2. Ela supostamente permaneceria em BAIXO.

Os efeitos do desalinhamento do clock nem sempre são fáceis de detectar, porque a resposta do FF afetado pode ser intermitente (algumas vezes funciona corretamente, outras não). Isto ocorre porque a situação depende dos atrasos de propagação do circuito e dos parâmetros de temporização do FF, que variam com a temperatura, com o comprimento das ligações, com a tensão de alimentação e com o carregamento. Algumas vezes, apenas a simples colocação da ponta de prova do osciloscópio em uma saída de FF ou porta lógica adiciona capacitância de carga suficiente para aumentar o atraso de propagação do dispositivo e fazer o circuito funcionar corretamente. Então, quando a ponta é removida do circuito, a operação incorreta reaparece. Este é o tipo de situação que explica por que alguns técnicos e engenheiros ficam grisalhos prematuramente.

Problemas causados por desalinhamento de clock podem ser eliminados igualando-se os atrasos das diversas trajetórias do sinal de clock, de modo que a transição ativa atinja cada FF aproximadamente ao mesmo tempo. Isto é examinado no Problema 5-49.

Questões de Revisão

1. O que é desalinhamento do clock? Como isto pode causar um problema?

RESUMO

1. Um flip-flop é um circuito lógico com propriedade de memória, de modo que suas saídas Q e $\overline{Q}$ vão para um novo estado, em resposta a um pulso de entrada, e permanecem neste novo estado após o pulso de entrada terminar.

2. Um latch NAND e um latch NOR são FFs simples que respondem a níveis lógicos nas suas entradas SET e CLEAR.

3. Limpar (ressetar) um FF significa que suas saídas assumem o estado $Q = 0/\overline{Q} = 1$. Setar um FF significa que suas saídas assumem o estado $Q = 1/\overline{Q} = 0$.

4. Flip-flops com clock têm uma entrada de clock (*CLK*, *CP*, *CK*) que é disparada pela borda, significando que ela dispara o FF na transição positiva ou na transição negativa.

5. Os FFs disparados pela borda podem ser levados para um novo estados pela transição ativa da entrada de clock, de acordo com os estados das entradas síncronas do FF (*S*, *C* ou *J*, *K* ou *D*).

6. A maioria dos FFs com clock também tem entradas assíncronas que podem setar ou ressetar o FF independentemente da entrada de clock.

7. O latch *D* é um latch NAND modificado que opera como um flip-flop D, só que ele não é disparado por transição.

8. Algumas das utilizações mais importantes dos FFs incluem armazenamento e transferência de dados, deslocamento serial de dados, contagem e divisão de freqüência.

9. Um monoestável é um circuito lógico que pode ser disparado a partir do seu estado normal de repouso ($Q = 0$) para seu estado ativo ($Q = 1$), onde permanece por um intervalo de tempo proporcional a uma constante de tempo *RC*.

10. Circuitos que possuem entradas do tipo Schmitt-trigger respondem de modo confiável a sinais que variam lentamente, e produzem saídas com transições bem-definidas.

11. Uma variedade de circuitos pode ser usada para gerar sinais de clock com uma freqüência desejada, dentre eles os osciladores Schmitt-trigger, temporizadores 555 e osciladores controlados a cristal.

12. Um resumo completo dos vários tipos de FFs pode ser encontrado no início deste livro.

TERMOS IMPORTANTES

flip-flop
estados/entradas de SET, CLEAR, RESET
latch com portas NAND
latch
trepidação de contato
latch com portas NOR
clock
transição positiva (subida)
transição negativa (descida)
flip-flop(s) com clock
disparado(a) por transição
entradas de controle síncronas
tempo de setup/tempo de hold
flip-flop S-C com clock
disparo (trigger)
latch NAND
circuito direcionador de pulsos
circuito detector de transição
flip-flop J-K com clock
modo de comutação
flip-flop D com clock
transferência paralela de dados
latch D
entradas assíncronas
entradas de sobreposição
bloco de controle comum
atraso(s) de propagação
flip-flop(s) mestre/escravo
registradores
transferência de dados
transferência síncrona
transferência assíncrona
transferência paralela
transferência serial
registrador de deslocamento
divisão de freqüência
contador binário
tabela de estados
diagrama de transição de estados
módulo
Schmitt-trigger
monoestável (MONO)
estado quase-estável
MONO(s) não-redisparável(eis)
MONO(s) redisparável(eis)
multivibrador astável
temporizador 555
gerador de clock a cristal
desalinhamento do clock

PROBLEMAS

SEÇÕES 5-1 A 5-3

5-1. Considerando que inicialmente $Q = 0$, aplique as formas de onda x e y da Fig. 5-62 nas entradas SET e CLEAR de um latch NAND, e determine as formas de onda de Q e $\overline{Q}$.

5-2. Inverta as formas de onda de x e y da Fig. 5-62, aplique-as nas entradas SET e CLEAR de um latch NOR e determine as formas de onda de Q e $\overline{Q}$. Considere que inicialmente $Q = 0$.

5-3. Suponha que as formas de onda da Fig. 5-62 estão conectadas ao circuito da Fig. 5-63. Considere que inicialmente $Q = 0$ e determine a forma de onda de Q.

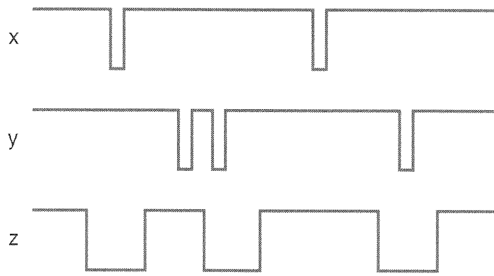


Fig. 5-62 Problemas 5-1 a 5-3.

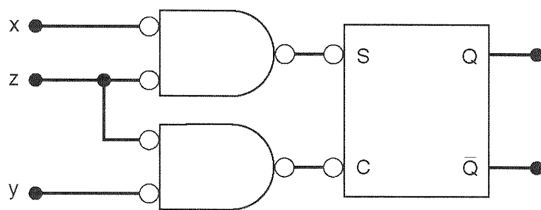


Fig. 5-63 Problema 5-3.

D

5-4. Modifique o circuito da Fig. 5-9 para usar um latch com portas NOR.

D

5-5. Modifique o circuito da Fig. 5-12 para usar um latch com portas NAND.

T

5-6. Observe o circuito da Fig. 5-13. Um estudante testa a operação do circuito visualizando as saídas em um osciloscópio com memória enquanto a chave é levada de A para B. Quando a chave vai de A para B, a forma de onda de X_B, tal como aparece na tela do osciloscópio, é mostrada na Fig. 5-64. Que falha de circuito poderia produzir este resultado? (*Sugestão:* Qual é a função do latch NAND?)

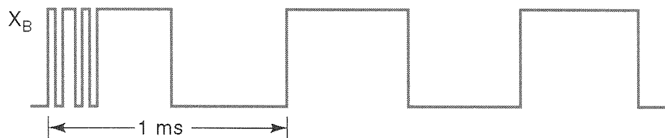


Fig. 5-64 Problema 5-6.

SEÇÕES 5-4 E 5-5

5-7. Um determinado FF com clock tem os seguintes tempos mínimos: $t_S = 20$ ns e $t_H = 5$ ns. Por quanto tempo as entradas de controle devem ficar estáveis antes da transição ativa do clock?

5-8. Aplique as formas de onda de S, C e CLK da Fig. 5-17 no FF da Fig. 5-18 e determine a forma de onda da saída Q.

N

5-9. Um flip-flop do tipo T (*toggle*) tem uma única entrada e opera de tal modo que a sua saída muda de estado para cada pulso aplicado à sua entrada. O flip-flop S-C com clock pode ser interligado para operar neste modo de comutação, conforme mostra a Fig. 5-65. A forma de onda aplicada à entrada CLK é uma onda quadrada de 1 kHz. Verifique que este arranjo opera no modo de comutação e então determine a forma de onda da saída Q. Considere que inicialmente $Q = 0$.

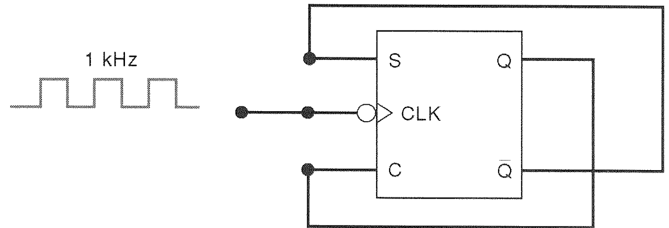


Fig. 5-65 Problema 5-9.

SEÇÃO 5-6

5-10. Aplique as formas de onda de J, K e CLK da Fig. 5-21 no FF da Fig. 5-22. Considere que inicialmente $Q = 1$ e determine a forma de onda da saída Q.

5-11. **(a)** Mostre como um flip-flop J-K pode funcionar como um FF do tipo T. Aplique uma onda quadrada de 10 kHz na sua entrada e determine sua forma de onda de saída.

(b) Conecte a saída Q do FF do Problema 5-11(a) na entrada CLK de um segundo flip-flop J-K que também tenha $J = K = 1$. Determine a freqüência da forma de onda de saída do segundo FF.

5-12. As formas de onda mostradas na Fig. 5-66 devem ser aplicadas em dois FFs diferentes:

(a) J-K disparado pela transição positiva

(b) J-K disparado pela transição negativa

Desenhe a forma de onda de Q para cada um dos FFs, supondo que inicialmente $Q = 0$. Considere que cada FF tem $t_H = 0$.

SEÇÃO 5-7

5-13. Um flip-flop D algumas vezes é usado para *atrasar* uma forma de onda binária de modo que a informação binária apareça na saída um certo tempo depois que apareceu na entrada.

(a) Determine a forma de onda de Q na Fig. 5-67 e compare-a com a forma de onda da entrada. Note que ela está atrasada por um período do clock em relação à entrada.

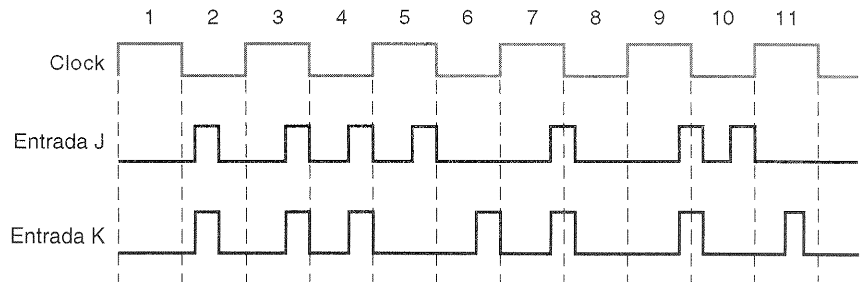


Fig. 5-66 Problema 5-12.

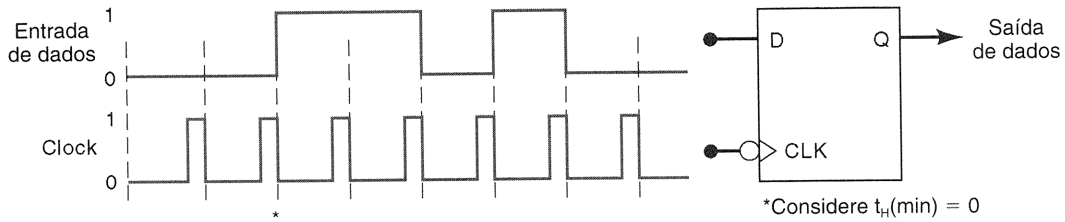


Fig. 5-67 Problema 5-13.

(b) Como se pode obter um atraso de dois períodos de clock?

5-14. Um flip-flop D disparado pela borda pode ser operado no modo de comutação conectando-o como mostra a Fig. 5-68. Considere que inicialmente $Q = 0$ e determine a forma de onda de Q.

5-15. Altere o circuito da Fig. 5-68 de modo que a saída Q seja ligada em D. Determine a forma de onda de Q.

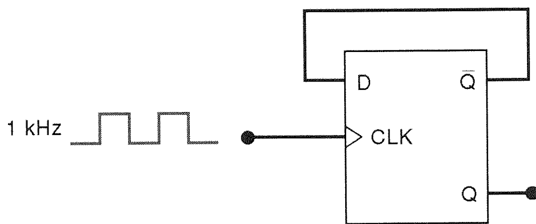


Fig. 5-68 Flip-flop D conectado para comutação (Problemas 5-14 e 5-15).

SEÇÃO 5-8

5-16. Compare a operação de um latch *D* com um flip-flop D disparado na transição negativa, aplicando as formas de onda da Fig. 5-69 a cada um e determinando as formas de onda de *Q*.

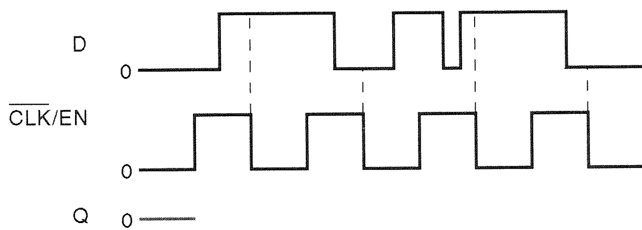


Fig. 5-69 Problema 5-16.

5-17. No Problema 5-14, vimos como um flip-flop D disparado pela borda pode ser operado no modo de comutação. Explique por que esta mesma idéia não funcionaria para um latch *D*.

SEÇÃO 5-9

5-18. Determine a forma de onda de *Q* para o FF na Fig. 5-70. Considere que inicialmente $Q = 0$, e lembre-se de que as entradas assíncronas sobrepõem-se a todas as outras entradas.

5-19. Aplique as formas de onda de $\overline{CLK}$, $\overline{PRE}$ e $\overline{CLR}$ da Fig. 5-31 em um flip-flop D disparado na transição positiva com entradas assíncronas ativas em BAIXO. Considere que *D* é mantido em ALTO e *Q* está inicialmente em BAIXO. Determine a forma de onda de *Q*.

5-20. Examine o símbolo IEEE/ANSI para o CI 74276 na Fig. 5-71.

(a) É possível setar ou ressetar flip-flops individuais assincronamente sem afetar os outros?

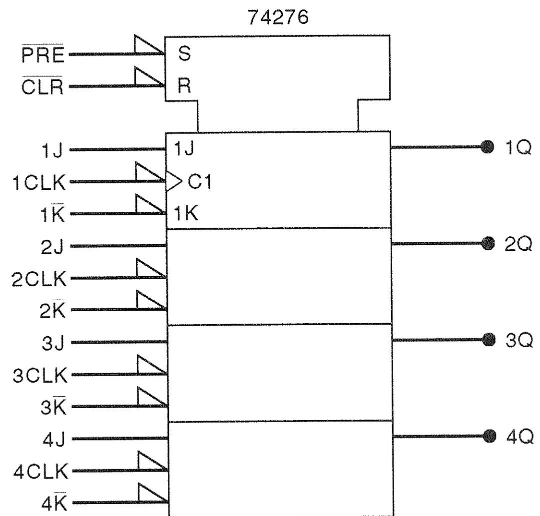


Fig. 5-71 Problema 5-20.

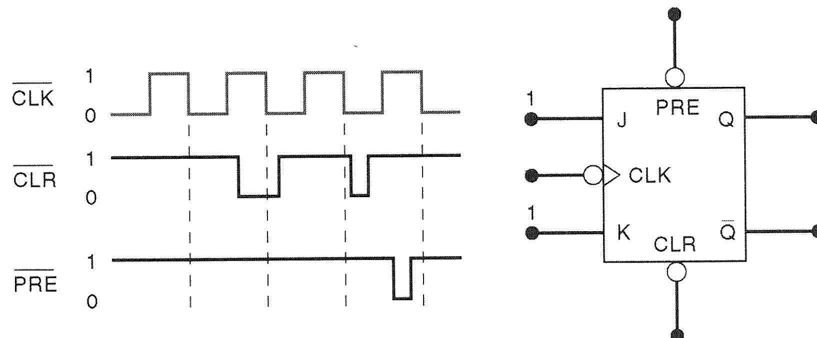


Fig. 5-70 Problema 5-18.

(b) Quais as condições de entrada necessárias para provocar a comutação da saída $1\overline{Q}$? (Note a barra nas entradas $\overline{K}$.)

SEÇÃO 5-11

5-21. Use a Tabela 5-2 na Seção 5-11 para determinar o seguinte:

(a) Quanto tempo pode levar a saída Q de um 74C74 para comutar de 0 para 1 em resposta a uma transição ativa do clock?

(b) Qual dos FFs na Tabela 5-2 necessita que suas entradas de controle permaneçam estáveis por mais tempo *depois* da transição ativa do clock? E *antes* da transição?

(c) Qual é o pulso mais estreito que pode ser aplicado ao $\overline{PRE}$ de um FF 7474?

5-22. Veja o circuito da Fig. 5-72. Ela mostra um CI 74LS112 com seus dois flip-flops J-K conectados de certo modo. Considere que inicialmente $Q_1 = Q_2 = 1$ e, usando a Tabela 5-2, determine o atraso *total* de propagação entre a descida do pulso de clock e a descida de Q_2.

SEÇÕES 5-15 E 5-16

D

5-23. Modifique o circuito da Fig. 5-40 para utilizar um flip-flop J-K.

D

5-24. No circuito da Fig. 5-73, todas as entradas A, B e C estão inicialmente em BAIXO. A saída Y deve ir para ALTO somente quando A, B e C forem para ALTO em uma certa seqüência.

(a) Determine a seqüência que faz Y ir para ALTO.

(b) Explique por que o pulso de START é necessário.

(c) Modifique este circuito para usar FFs do tipo D.

SEÇÕES 5-17 E 5-18

D

5-25. (a) Desenhe um diagrama de circuito para a transferência de dados paralela e síncrona de um registrador de três bits para outro usando flip-flops J-K.

(b) Repita para transferência paralela assíncrona.

5-26. Um registrador de deslocamento *circular* é um registrador de deslocamento que mantém a informação binária circulando através do registrador conforme os pulsos de clock são aplicados. O registrador de deslocamento da Fig. 5-45 pode ser transformado em um registrador circular conectando-se X_0 na linha DATA IN. Nenhuma entrada externa é usada. Considere que este registrador comece com 1011 armazenado nele (isto é, $X_3 = 1$, $X_2 = 0$, $X_1 = 1$ e $X_0 = 1$). Relacione a seqüência de estados que os FFs do registra-

Fig. 5-72 Diagrama de conexão para o Problema 5-22.

Fig. 5-73 Problema 5-24.

dor apresentam conforme oito pulsos de deslocamento são aplicados.

D

5-27. Observe a Fig. 5-46, onde um número de três bits armazenado no registrador *X* é deslocado serialmente para o registrador *Y*. Como o circuito poderia ser modificado para que, ao final da transferência, o valor do número original armazenado em *X* estivesse presente em ambos os registradores? (*Sugestão:* Veja o Problema 5-26.)

SEÇÃO 5-19

5-28. Observe o contador binário da Fig. 5-47. Modifique-o conectando $\overline{X_0}$ ao *CLK* do flip-flop X_1, e $\overline{X_1}$ ao *CLK* do flip-flop X_2. Comece com todos os FFs no estado 1 e desenhe as formas de onda de saída (X_0, X_1 e X_2) para 16 pulsos de entrada. Depois relacione a seqüência de estados dos FFs como foi feito na Fig. 5-48. Este contador é chamado de contador *decrescente*. Por quê?

5-29. Desenhe o diagrama de transição de estados para o contador decrescente e compare-o com o diagrama da Fig. 5-49. Em que eles são diferentes?

D

5-30. Mostre como os flip-flops D com clock podem ser usados em um contador como o da Fig. 5-47. (*Sugestão:* Veja o Problema 5-14.)

5-31. **(a)** Quantos FFs são necessários para construir um contador binário capaz de contar desde 0 até 1.023?
(b) Determine a freqüência na saída do último FF deste contador para um clock de entrada com freqüência de 2 MHz.
(c) O que é o módulo de um contador?
(d) Se o contador está inicialmente com zero, que valor de contagem ele apresentará depois de 2.060 pulsos?

5-32. Um contador binário está sendo acionado por um sinal de clock de 256 kHz. A freqüência de saída do último FF é 2 kHz.
(a) Determine o módulo.
(b) Determine a faixa de contagem.

5-33. Um circuito fotodetector está sendo usado para gerar um pulso cada vez que um cliente entra em determinado estabelecimento. Os pulsos são levados para um contador de oito bits. O contador é usado para contar estes pulsos de modo a determinar quantos clientes entraram na loja. Depois de fechar a loja, o proprietário verifica o contador e encontra o valor $00001001_2 = 9_{10}$. Ele sabe que isto está errado, pois muito mais do que nove pessoas entraram na loja. Considerando que o circuito contador esteja operando adequadamente, qual poderia ser a razão para a discrepância?

SEÇÃO 5-20

D

5-34. Modifique o circuito da Fig. 5-50 para que apenas a presença do endereço 10110110 permita que o dado seja transferido para o registrador *X*.

T

5-35. Suponha que o circuito da Fig. 5-50 esteja com problemas de funcionamento, de modo que dados estão sendo transferidos para *X* tanto pelo endereço 11111110 quanto pelo 11111111. Quais são algumas das falhas de circuito que poderiam causar isto?

D

5-36. Modifique o circuito da Fig. 5-50 para que a MPU tenha oito bits de dados de saída conectados para transferir dados de oito bits para um registrador de oito bits construído com dois CIs 74HC175 [Fig. 5-34(b)]. Mostre todas as conexões do circuito.

SEÇÃO 5-22

N

5-37. A Fig. 5-74 mostra três monoestáveis não-redisparáveis que produzem três pulsos de saída seqüenciais. Note o "1" na frente de cada pulso dentro dos símbolos dos monoestáveis para indicar operação não-redisparável. Desenhe um diagrama temporal que mostra a relação entre o pulso de entrada e as três saídas dos monoestáveis. Considere uma duração de 10 ms para o pulso de entrada.

5-38. Um monoestável *redisparável* pode ser usado como um detector de freqüência de pulsos que detecta quando a freqüência dos pulsos de entrada está abaixo de um valor predeterminado. Um exemplo simples desta aplicação é mostrado na Fig. 5-75. A operação é iniciada fechando-se a chave SW1 momentaneamente.
(a) Descreva como o circuito responde a freqüências de entrada acima de 1 kHz.
(b) Descreva como o circuito responde a freqüências de entrada abaixo de 1 kHz.
(c) Como você modificaria o circuito para detectar quando a freqüência de entrada caísse abaixo de 50 kHz?

5-39. Consulte o símbolo lógico para o monoestável não-redisparável 74121 na Fig. 5-54(a).
(a) Que condições de entrada são necessárias para o MONO disparar por um sinal na entrada *B*?
(b) Que condições de entrada são necessárias para o MONO disparar por um sinal na entrada A_1?

D, C

5-40. A largura do pulso de saída de um monoestável 74121 é dada pela fórmula aproximada

$$t_p \approx 0{,}7\ R_T C_T$$

onde R_T é a resistência conectada entre os pinos R_{EXT}/C_{EXT} e V_{CC} e C_T é a capacitância conectada entre os pinos C_{EXT} e R_{EXT}/C_{EXT}. O valor de R_T pode variar entre 2 e 40 kΩ, e C_T pode ser tão grande quanto 1.000 μF.
(a) Mostre como um 74121 pode ser conectado para produzir um pulso de saída ativo em BAIXO com uma duração de 5 ms, sempre que um dos dois sinais lógicos (*E* ou *F*) fizer uma transição negativa. Tanto *E* quanto *F* estão normalmente no estado ALTO.
(b) Modifique o circuito de modo que um sinal de controle, *G*, possa desabilitar o pulso de saída do monoestável independentemente do que ocorrer em *E* e *F*.

SEÇÃO 5-23

C

5-41. Considere o circuito da Fig. 5-76. Inicialmente todos os FFs estão no estado 0. A operação do circuito começa com um

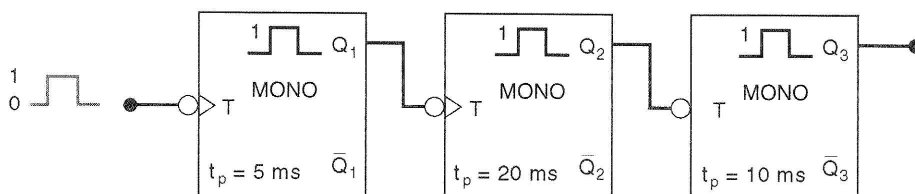


Fig. 5-74 Problema 5-37.

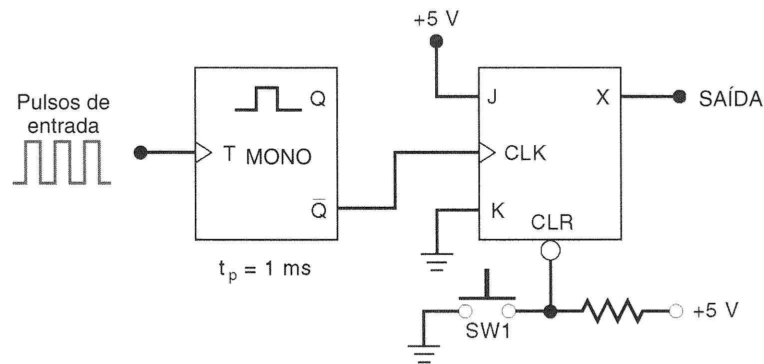


Fig. 5-75 Problema 5-38.

pulso de início aplicado na entrada $\overline{PRESET}$ dos FFs *X* e *Y*. Determine as formas de onda em *A*, *B*, *C*, *X*, *Y*, *Z* e *W* para 20 ciclos do clock depois do pulso de início. Apresente todas as suas suposições.

SEÇÃO 5-24

5-42. Mostre como usar um INVERSOR Schmitt-trigger para produzir uma forma de onda quadrada aproximada com freqüência de 10 kHz.

5-43. Projete um oscilador com 555 para produzir uma forma de onda quadrada aproximada com 40 kHz. *C* deve ser maior ou igual a 500 pF.

5-44. Um oscilador com 555 pode ser combinado com um flip-flop J-K para produzir uma onda quadrada perfeita (taxa de ciclo de 50%). Modifique o circuito do Problema 5-43 para incluir um flip-flop J-K. A saída final deve ser ainda uma onda quadrada de 40 kHz.

C, N

5-45. O circuito da Fig. 5-77 pode ser usado para gerar dois sinais de clock não-sobrepostos e de mesma freqüência. Estes sinais de clock são usados em alguns sistemas de microprocessadores que necessitam de quatro diferentes transições do clock para sincronizar suas operações.

(a) Desenhe as formas de onda temporais de CP1 e CP2 se o *CLOCK* é uma onda quadrada de 1 MHz. Considere que t_{PLH} e t_{PHL} são iguais a 20 ns para o FF e 10 ns para as portas AND.

(b) Este circuito teria um problema se o FF fosse trocado por um outro com transição positiva do *CLK*. Desenhe as formas de onda de CP1 e CP2 para esta situação. Preste atenção especial a condições que possam produzir glitches.

SEÇÃO 5-25

T

5-46. Observe o circuito contador da Fig. 5-47. Considere que todas as entradas assíncronas estão conectadas em V_{CC}. Quando foi testado, o circuito apresentou as formas de onda ilustradas na Fig. 5-78. Considere a seguinte lista de falhas possíveis. Para cada uma, indique "sim" ou "não" caso ela possa ser a causa dos resultados observados. Explique cada resposta.

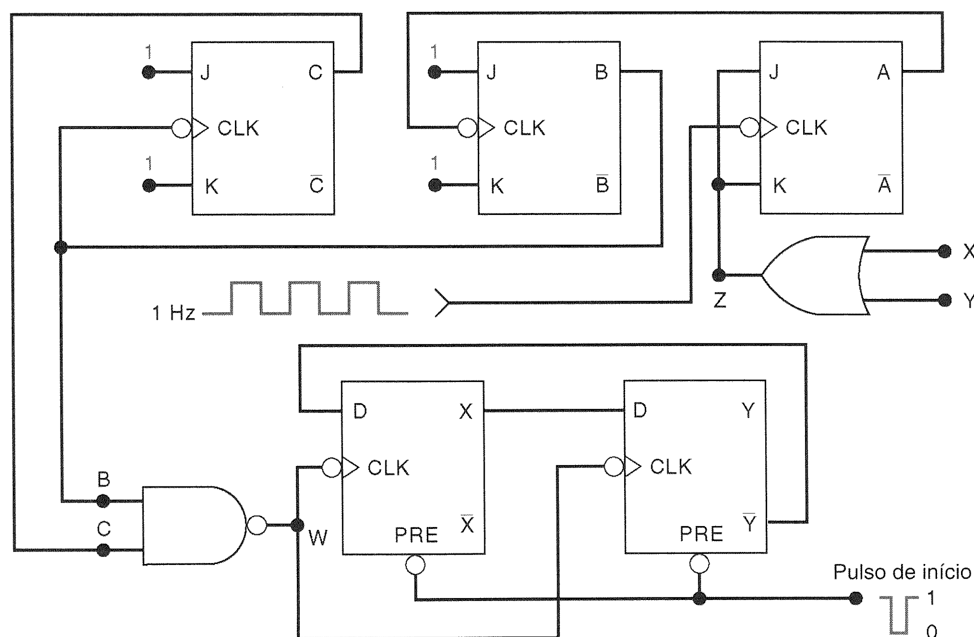


Fig. 5-76 Problema 5-41.

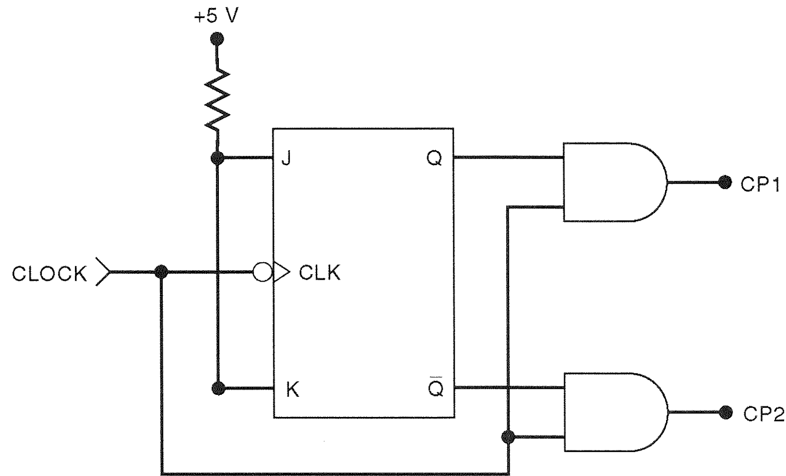


Fig. 5-77 Problema 5-45.

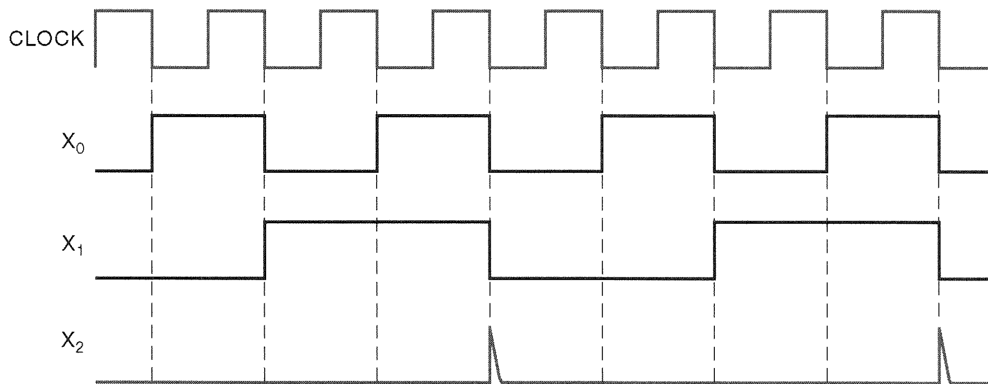


Fig. 5-78 Problema 5-46.

(a) A entrada *CLR* de X_2 está aberta.
(b) O tempo de transição da saída X_1 está muito grande, possivelmente devido ao carregamento.
(c) A saída X_2 está em curto com a terra.
(d) O tempo de hold de X_2 não está sendo respeitado.

T
5-47. Consulte o circuito da Fig. 5-46. Todos os FFs são CIs TTL. Considere as seguintes condições iniciais: $X_2X_1X_0 = 100$ e $Y_2Y_1Y_0 = 011$. Após quatro pulsos de deslocamento, as condições são: $X_2X_1X_0 = 001$ e $Y_2Y_1Y_0 = 111$. Pulsos de deslocamento subseqüentes não provocam mudança alguma em nenhum FF. Quais são as possíveis causas deste mau funcionamento?

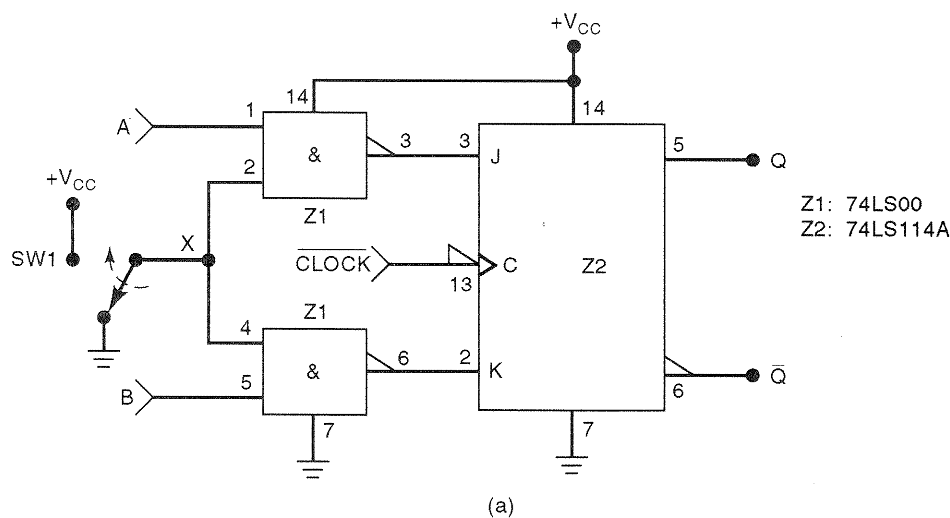
C, T
5-48. Considere a situação da Fig. 5-61 para cada um dos seguintes conjuntos de parâmetros de tempo. Para cada um deles indique se o flip-flop Q_2 responderá corretamente.
(a) Cada FF: t_{PLH} = 12 ns; t_{PHL} = 8 ns; t_S = 5 ns; t_H = 0 ns
Porta NAND: t_{PLH} = 8 ns; t_{PHL} = 6 ns
INVERSOR: t_{PLH} = 7 ns; t_{PHL} = 5 ns
(b) Cada FF: t_{PLH} = 10 ns; t_{PHL} = 8 ns; t_S = 5 ns; t_H = 0 ns
Porta NAND: t_{PLH} = 12 ns; t_{PHL} = 10 ns
INVERSOR: t_{PLH} = 8 ns; t_{PHL} = 6 ns

D
5-49. Mostre e explique como o problema de desalinhamento do clock da Fig. 5-61 pode ser eliminado pela inserção apropriada de dois INVERSOREs.

T
5-50. Consulte o circuito da Fig. 5-55. Descreva como a operação do circuito mudará para cada uma das seguintes falhas.
(a) Um curto-circuito interno com a terra na entrada superior da porta NAND
(b) Uma conexão aberta na entrada *J* do FF *Z*
(c) Uma conexão aberta na entrada inferior da porta NAND

T
5-51. Veja o circuito da Fig. 5-79. Considere que os CIs são da família lógica TTL. A forma de onda de *Q* foi obtida quando o circuito foi testado com os sinais de entrada mostrados e com a chave na posição de cima. Ela não está correta. Considere a seguinte lista de falhas, e para cada uma delas indique "sim" ou "não" caso ela possa ser a falha real. Explique cada resposta.
(a) O ponto X está sempre em BAIXO devido a uma chave defeituosa.
(b) O pino 1 de Z1 está internamente em curto-circuito com V_{CC}.
(c) A conexão entre Z1-3 e Z2-3 está interrompida.
(d) Existe uma ponte de solda entre os pinos 6 e 7 de Z1.

C
5-52. O circuito da Fig. 5-80 funciona como uma tranca de combinação seqüencial. Para abrir a tranca, proceda da seguinte maneira:
1. Ative momentaneamente a chave de INÍCIO.
2. Ajuste as chaves SWA, SWB e SWC para a primeira parte



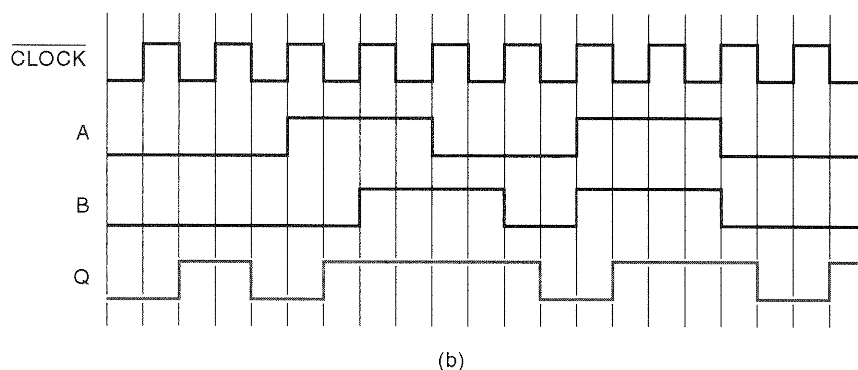


Fig. 5-79 Problema 5-51.

do segredo. Então comute a chave ENTER momentaneamente.

3. Ajuste as chaves para a segunda parte do segredo e comute ENTER novamente. Isto deve produzir um nível ALTO em Q_2 para abrir a tranca.
Se a combinação incorreta for apresentada em qualquer dos passos, o operador deve reiniciar a seqüência novamente. Analise o circuito e determine a seqüência de valores que abre a tranca.

C, T

5-53. Quando a tranca da Fig. 5-80 foi testada, constatou-se que fornecendo a seqüência correta ela não abria. Um teste com uma ponta de prova lógica mostrou que, entrando com a primeira combinação correta, Q_1 é setado em ALTO, mas entrando com a segunda combinação produz-se apenas um pulso em Q_2. Considere cada uma das falhas seguintes e indique quais delas poderiam produzir a operação observada. Explique cada escolha.
(a) Efeito da trepidação de contato em SWA, SWB e SWC.
(b) A entrada *CLR* de Q_2 está aberta.
(c) A conexão da saída da porta NAND 4 para a entrada da porta NAND 3 está aberta.

QUESTÕES DE FIXAÇÃO

5-54. Para cada afirmação indique que tipo de FF está sendo descrito.
(a) Tem entradas de SET e CLEAR mas não tem uma entrada *CLK*.
(b) Comuta a cada pulso em *CLK* quando suas entradas de controle estão ambas em ALTO.
(c) Tem uma entrada de ENABLE em vez de uma entrada *CLK*.
(d) É usado para transferir dados facilmente de um registrador para outro.
(e) Tem apenas uma entrada de controle.
(f) Tem duas saídas que são complementares entre si.
(g) Pode mudar de estado somente na transição ativa de *CLK*.
(h) É usado em contadores binários.

5-55. Defina os seguintes termos.
(a) Entradas assíncronas
(b) Disparado pela borda
(c) Registrador de deslocamento
(d) Divisão de freqüência
(e) Transferência assíncrona
(f) Diagrama de transição de estados
(g) Transferência paralela de dados
(h) Transferência serial de dados
(i) Monoestável redisparável
(j) Entradas Schmitt-trigger

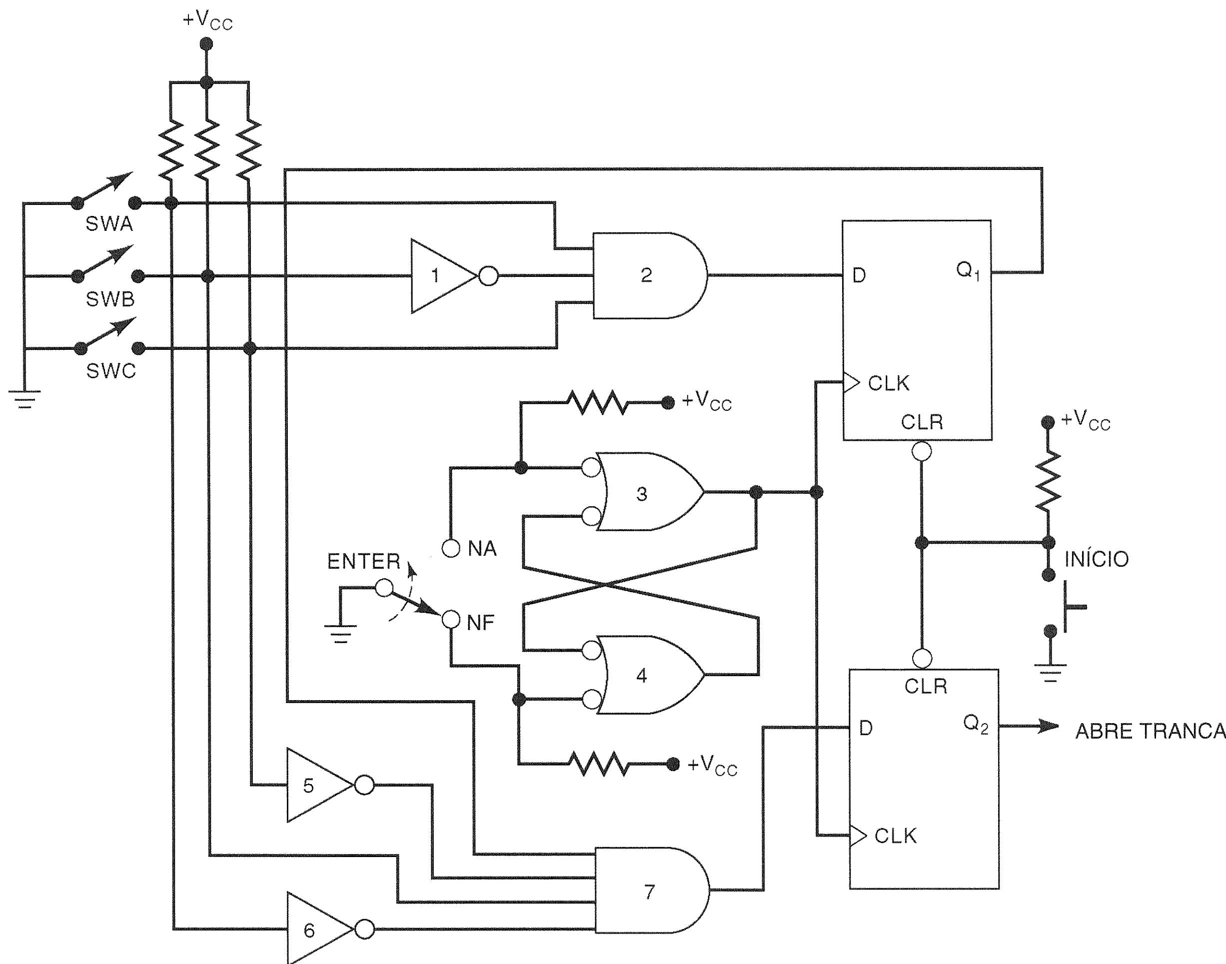


Fig. 5-80 Problemas 5-52 e 5-53.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 5-1

1. ALTO; BAIXO **2.** $Q = 0$, $\overline{Q} = 1$ **3.** Verdadeiro
4. Aplicar um nível BAIXO momentaneamente na entrada $\overline{SET}$.

SEÇÃO 5-2

1. BAIXO, ALTO **2.** $Q = 1$, $\overline{Q} = 0$ **3.** Fazer CLEAR = 1
4. $\overline{SET}$ e $\overline{CLEAR}$ estariam ambas normalmente no seu estado ativo em BAIXO.

SEÇÃO 5-4

1. Entradas de controle síncronas e entrada de clock.
2. A saída do FF pode mudar somente quando a transição apropriada do clock ocorrer. **3.** Falso. **4.** Tempo de setup é o intervalo de tempo imediatamente anterior à transição ativa do sinal *CLK*, durante o qual as entradas de controle devem permanecer estáveis. Tempo de hold é o intervalo de tempo imediatamente após a transição ativa de *CLK*, durante o qual as entradas de controle devem permanecer estáveis.

SEÇÃO 5-5

1. Irá para ALTO. **2.** Porque CLK* está ALTO somente por alguns nanossegundos.

SEÇÃO 5-6

1. Verdadeiro **2.** Não **3.** $J = 1$, $K = 0$

SEÇÃO 5-7

1. *Q* vai para BAIXO no ponto *a* e permanece BAIXO.
2. Falso. A entrada *D* pode mudar sem afetar *Q*, pois *Q* só pode mudar na borda ativa de *CLK*.
3. Sim

SEÇÃO 5-8

1. Em um latch *D* a saída *Q* pode mudar enquanto *EN* está ALTO. Em um flip-flop D, a saída só pode mudar na borda ativa de *CLK*.
2. Falso. **3.** Verdadeiro.

SEÇÃO 5-9

1. Entradas assíncronas operam independentemente da entrada *CLK*. **2.** Sim, já que $\overline{PRE}$ é ativo em BAIXO.
3. $J = K = 1$, $\overline{PRE} = \overline{CLR} = 1$ e uma transição positiva em *CLK*.

SEÇÃO 5-10

1. O triângulo dentro do retângulo indica operação por transição; o triângulo reto externo ao retângulo indica disparo na descida.
2. É usado para indicar que a função dessas entradas é comum a mais de um circuito no chip.

SEÇÃO 5-11
1. t_{PLH} e t_{PHL}. **2.** Falso, pois a forma de onda também deve satisfazer $t_W(L)$ e $t_W(H)$.

SEÇÃO 5-17
1. Falso **2.** Flip-flop D **3.** Seis **4.** Verdadeiro

SEÇÃO 5-18
1. Verdadeiro **2.** Poucas interconexões entre registradores
3. $X_2X_1X_0 = 111$; $Y_2Y_1Y_0 = 101$ **4.** Paralela

SEÇÃO 5-19
1. 10 kHz **2.** Oito **3.** 256 **4.** 2 kHz
5. $00001000_2 = 8_{10}$

SEÇÃO 5-21
1. A saída pode conter oscilações. **2.** Ele produz sinais de saída limpos e rápidos mesmo para entradas que variam lentamente.

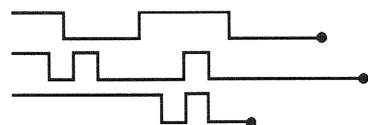
SEÇÃO 5-22
1. $Q = 0$, $\overline{Q} = 1$ **2.** Verdadeiro **3.** Os valores de *R* e *C* externos
4. Para um monoestável redisparável, cada novo pulso de disparo inicia um novo intervalo t_p, não importando o estado da saída *Q*.

SEÇÃO 5-24
1. 24 kHz **2.** 109,3 kHz; 66,7 por cento
3. Estabilidade da freqüência

SEÇÃO 5-25
1. O desalinhamento do clock é a chegada de um sinal de clock nas entradas *CLK* de diferentes FFs em instantes de tempo diferentes. Ele pode causar que um FF vá para um estado incorreto.

CAPÍTULO 6

Aritmética Digital: Operações e Circuitos



SUMÁRIO

OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Efetuar soma, subtração, multiplicação e divisão com dois números binários.
- Somar e subtrair números hexadecimais.
- Saber a diferença entre a soma binária e a soma OR.
- Comparar as vantagens e desvantagens entre três sistemas diferentes de representação de números binários com sinal.
- Manipular números binários com sinal utilizando o sistema de complemento a 2.
- Compreender o circuito somador BCD e o processo de soma BCD.
- Descrever a operação básica de uma unidade lógica e aritmética.
- Utilizar somadores completos no projeto de somadores binários paralelos.
- Citar as vantagens dos somadores paralelos com carry antecipado.
- Explicar a operação do circuito somador/subtrator paralelo.
- Utilizar um circuito integrado de ULA para realizar várias operações lógicas e aritméticas sobre os dados de entrada.
- Ler e entender o símbolo IEEE/ANSI de um somador paralelo.
- Analisar os estudos de caso de pesquisa de falhas de circuitos somadores/subtratores.

INTRODUÇÃO

Computadores digitais e calculadoras realizam as várias operações aritméticas sobre os números representados em forma binária. O tópico de aritmética digital pode ser bastante complexo se desejarmos entender todos os métodos computacionais e a teoria por trás deles. Felizmente, este nível de conhecimento não é necessário para a maioria dos estudantes, pelo menos até se tornarem programadores experientes. Nossa abordagem neste capítulo se concentrará nos princípios básicos necessários para entender como as máquinas digitais (isto é, os computadores) realizam as operações aritméticas básicas.

Primeiramente veremos como as diversas operações aritméticas são feitas com números binários usando-se "lápis e papel", e depois estudaremos os circuitos lógicos reais que realizam estas operações em um sistema digital.

6-1 ADIÇÃO BINÁRIA

A adição de dois números binários é realizada da mesma maneira que a adição de números decimais. De fato, a adição binária é mais simples, já que existem menos casos a aprender. Primeiramente vamos rever a adição decimal:

```
      ┌────┐
      ↓    ↓
 3  7  6   LSD
+4  6  1
--------
 8  3  7
```

A posição do dígito menos significativo (LSD) é efetuada primeiro, produzindo uma soma de 7. Os dígitos da segunda posição são somados em seguida e produzem uma soma de 13, que gera um **carry** (vai-um) de valor 1 para a terceira posição. Isto resulta em uma soma de 8 na terceira posição.

Os mesmos passos gerais são seguidos na adição binária. Entretanto, apenas quatro situações podem ocorrer quando dois dígitos binários (bits) são somados, qualquer que seja a posição. Elas são:

$0 + 0 = 0$
$1 + 0 = 1$
$1 + 1 = 10 = 0$ +carry 1 para a próxima posição
$1 + 1 + 1 = 11 = 1$ +carry 1 para a próxima posição

O último caso ocorre quando os dois bits de uma certa posição são iguais a 1 e existe o carry da posição anterior. A seguir, diversos exemplos de adição de dois números binários (os equivalentes decimais estão entre parênteses):

```
   011 (3)       1001 (9)       11,011 (3,375)
 + 110 (6)     + 1111 (15)    + 10,110 (2,750)
 ---------     -----------    ----------------
  1001 (9)      11000 (24)     110,001 (6,125)
```

Não é necessário abordar a adição de mais do que dois números binários de uma vez, pois em todos os sistemas digitais o circuito que realmente realiza a adição pode tratar apenas dois números de cada vez. Quando mais de dois números devem ser somados, os dois primeiros são somados, e então a soma resultante é adicionada ao terceiro número, e assim por diante. Isto não é uma desvantagem séria, tendo em vista que os modernos computadores digitais podem realizar, geralmente, uma operação de soma em muitos nanossegundos.

A adição é a operação aritmética mais importante nos sistemas digitais. Conforme veremos, as operações de subtração, multiplicação e divisão, do modo que são realizadas na maioria dos computadores digitais e calculadoras, na verdade utilizam apenas a adição como sua operação básica.

Questões de Revisão

1. Some os seguintes pares de números binários.

(a) 10110 + 00111 **(b)** 011,101 + 010,010
(c) 10001111 + 00000001

6-2 REPRESENTAÇÃO DE NÚMEROS COM SINAL

Nos computadores digitais, os números binários são representados por um conjunto de dispositivos de armazenamen-

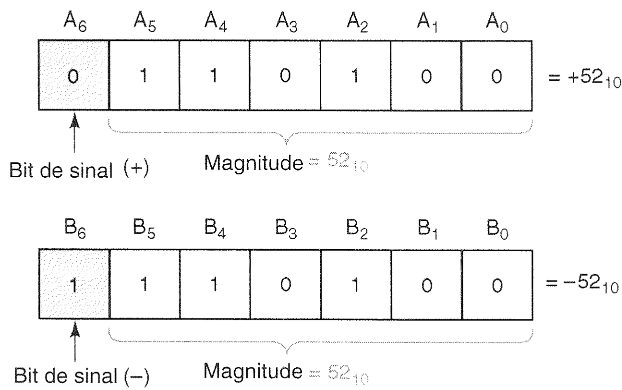


Fig. 6-1 - Representação de números com sinal na forma sinal-magnitude.

to binário (usualmente flip-flops). Cada dispositivo representa um bit. Por exemplo, um registrador com seis FFs poderia armazenar números binários desde 000000 até 111111 (0 a 63 em decimal). Isto representa a *magnitude* do número. Como a maioria dos computadores digitais e calculadoras opera tanto com números negativos quanto com números positivos, algum modo para representar o *sinal* do número (+ ou −) é necessário. Isto usualmente é feito incluindo-se ao número um outro bit denominado **bit de sinal**. Geralmente, a convenção comum que tem sido adotada é a de que um 0 no bit de sinal representa um número positivo, e um 1 no bit de sinal representa um número negativo. Isto é ilustrado na Fig. 6-1. O registrador A contém os bits 0110100. O 0 no bit mais à esquerda (A_6) é o bit de sinal que representa +. Os outros seis bits são a magnitude do número 110100_2, que é igual a 52 em decimal. Logo, o número armazenado no registrador A é +52. Analogamente, o número armazenado no registrador B é −52, pois o bit de sinal é 1, representando −.

O bit de sinal é usado para indicar a natureza positiva ou negativa do número binário armazenado. Os números na Fig. 6-1 consistem em um bit de sinal e seis bits de magnitude. Os bits de magnitude são o verdadeiro equivalente binário do valor decimal representado. Isto é chamado de **sistema sinal-magnitude** para representação de números binários com sinal.

Embora o sistema sinal-magnitude seja direto, calculadoras e computadores não o utilizam normalmente porque a implementação do circuito é mais complexa do que em outros sistemas. O sistema mais amplamente utilizado para representação de números binários com sinal é o **sistema de complemento a 2**. Antes de apresentarmos como isto é feito, devemos primeiramente ver como formar o complemento a 1 e o complemento a 2 de um número binário.

Forma do Complemento a 1

O complemento a 1 de um número binário é obtido substituindo-se cada 0 por 1 e cada 1 por 0. Em outras palavras, substitui-se cada bit no número binário pelo seu complemento. O processo é ilustrado a seguir.

1 0 1 1 0 1	número binário original
↓ ↓ ↓ ↓ ↓ ↓	complementa-se cada bit para formar
0 1 0 0 1 0	o complemento a 1

Deste modo, diz-se que o complemento a 1 de 101101 é 010010.

Forma do Complemento a 2

O complemento a 2 de um número binário é formado tomando-se o complemento a 1 do número e adicionando-se 1 na posição do bit menos significativo. O processo é ilustrado a seguir para $101101_2 = 45_{10}$.

1 0 1 1 0 1	equivalente binário de 45
0 1 0 0 1 0	complementa-se cada bit para formar o complemento a 1
+ 1	adiciona-se 1 para formar o complemento a 2
0 1 0 0 1 1	complemento a 2 do número binário original

Logo, diz-se que 010011 é a representação em complemento a 2 de 101101.

Segue um outro exemplo de conversão de um número binário para sua representação em complemento a 2:

1 0 1 1 0 0	número binário original
0 1 0 0 1 1	complemento a 1
+ 1	soma 1
0 1 0 1 0 0	complemento a 2 do número original

Representação de Números com Sinal Usando Complemento a 2

O sistema de complemento a 2 para representar números com sinal funciona do seguinte modo:

- Se o número é positivo, a magnitude é representada na sua forma binária direta, e um bit de sinal 0 é colocado na frente do MSB. Isto é mostrado na Fig. 6-2 para $+45_{10}$.
- Se o número é negativo, a magnitude é representada na sua forma de complemento a 2, e um bit de sinal 1 é colocado na frente do MSB. Isto é ilustrado na Fig. 6-2 para -45_{10}.

O sistema de complemento a 2 é usado para representar números com sinal porque, conforme veremos, ele permite realizar a operação de subtração efetuando na verdade uma adição. Isto é importante, pois significa que um computador digital pode usar os mesmos circuitos tanto para somar como para subtrair, deste modo economizando em hardware.

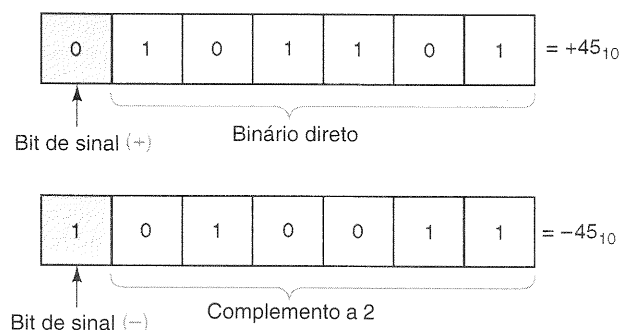


Fig. 6-2 - Representação de números com sinal no sistema de complemento a 2.

EXEMPLO 6-1

Represente cada um dos seguintes números decimais com sinal como um número binário com sinal no sistema de complemento a 2. Utilize cinco bits no total incluindo o bit de sinal.

(a) +13 **(b)** −9 **(c)** +3 **(d)** −2 **(e)** −8

Solução

(a) Como o número é positivo, a magnitude (13) é representada por sua magnitude direta, isto é, $13 = 1101_2$. Incluindo-se 0 como bit de sinal, obtém-se

+13 = 01101
bit de sinal ⮥

(b) O número é negativo, portanto a magnitude (9) deve ser representada na forma de complemento a 2:

$9_{10} = 1001_2$
0110 (complemento a 1)
+ 1 (soma 1 ao LSB)
0111 (complemento a 2)

Quando o bit de sinal é acrescentado, o número completo se torna

−9 = 10111
bit de sinal ⮥

O procedimento seguido necessita de dois passos. Primeiro determina-se o complemento a 2 da magnitude e então acrescenta-se o bit de sinal. Isto pode ser feito em um passo se incluirmos o bit de sinal na geração do complemento a 2. Por exemplo, para achar a representação para −9, começamos com a representação para +9, *incluindo o bit de sinal*, e calculamos o complemento a 2 dele de modo a obter a representação para −9.

+9 = 01001
10110 (complemento a 1 de cada bit incluindo o bit de sinal)
+ 1 (soma 1 ao LSB)
10111 (representação em complemento a 2 de −9)

O resultado é, obviamente, o mesmo que antes.

(c) O valor decimal 3 pode ser representado em binário usando-se apenas dois bits. Entretanto, o problema requer uma magnitude de quatro bits precedida por um bit de sinal. Portanto, temos

$+3_{10}$ = **00011**

Em várias situações o número de bits é fixado pelo tamanho dos registradores que armazenarão os números binários, de modo que 0s podem ter que ser incluídos para completar o número de bits necessários.

(d) Inicie escrevendo +2 usando cinco bits:

+2 = 00010
11101 (complemento a 1)
+ 1 (soma 1)
11110 (representação em complemento a 2 de −2)

(e) Comece com +8:

+8 = 01000
10111 (complementar cada bit)
+ 1 (somar 1)
11000 (representação em complemento a 2 de −8)

Negação

Negação é a operação de converter um número positivo no seu negativo equivalente ou um número negativo no seu positivo equivalente. Quando números binários com sinal são representados no sistema de complemento a 2, a negação é realizada simplesmente efetuando-se a operação de complemento a 2. Para ilustrar, considere o número +9. A sua representação com sinal é 01001. Se aplicarmos o complemento a 2, obtemos 10111. Certamente, isto é um número negativo já que o bit de sinal é 1. De fato, 10111 representa −9, que é o equivalente negativo do número original. Analogamente, podemos iniciar com a representação para −9, que é 10111. Tomando o complemento a 2, obtemos 01001, que reconhecemos como sendo +9. Esses passos estão relacionados a seguir.

iniciar com → 01001 = +9
fazer o complemento a 2 (negar) → 10111 = −9
negar novamente → 01001 = +9

Portanto, negamos um número binário com sinal obtendo-se o seu complemento a 2.

Essa negação altera o número para seu equivalente de sinal oposto. Utilizamos negação nos passos (d) e (e) do Exemplo 6-1 para converter números positivos nos seus equivalentes negativos.

EXEMPLO 6-2

Cada um dos números a seguir é um número binário com sinal no sistema de complemento a 2. Determine o valor decimal de cada um:

(a) 01100 **(b)** 11010 **(c)** 10001

Solução

(a) O bit de sinal é 0, portanto o número é *positivo* e os outros quatro bits representam a magnitude do número. Isto é, $1100_2 = 12_{10}$. Logo, o número decimal é **+12**.

(b) O bit de sinal de 11010 é 1, portanto sabemos que é um número negativo, mas não sabemos de que magnitude. Podemos determinar a magnitude negando (efetuando o complemento a 2) o número para convertê-lo no seu equivalente positivo.

11010 (número negativo original)
00101 (complemento a 1)
+ 1 (soma 1)
00110 (+6)

Tendo em vista que o resultado da negação é 00110 = +6, o número original 11010 deve ser equivalente a **−6**.

(c) Siga o mesmo procedimento do item (b):

```
   10001 (número negativo original)
   01110 (complemento a 1)
+      1 (soma 1)
   -----
   01111     (+15)
```

Logo, 10001 = **−15**.

Caso Especial na Representação de Complemento a 2

Sempre que um número com sinal tem 1 como bit de sinal e 0s para todos os bits de magnitude, seu equivalente decimal é -2^N, onde N é o número de bits na *magnitude*. Por exemplo,

$$1000 = -2^3 = -8$$
$$10000 = -2^4 = -16$$
$$100000 = -2^5 = -32$$

e assim por diante.

Portanto, podemos afirmar que a faixa completa de valores que pode ser representada no sistema de complemento a 2 que tem N bits de magnitude é

$$-2^N \text{ até } +(2^N - 1)$$

Existe um total de 2^{N+1} valores diferentes, *incluindo o zero*.

Por exemplo, a Tabela 6-1 relaciona todos os números com sinal que podem ser representados com quatro bits usando o sistema de complemento a 2 (note que existem três bits de magnitude, portanto $N = 3$). Repare que a seqüência começa em $-2^N = -2^3 = -8_{10} = 1000_2$ e continua de modo crescente até $+(2^N - 1) = +2^3 - 1 = +7_{10} = 0111_2$ somando-se 0001 em cada passo, tal como em um contador crescente.

TABELA 6-1

Valor Decimal	Binário com Sinal Utilizando Complemento a 2
$+7 = 2^3 - 1$	0111
+6	0110
+5	0101
+4	0100
+3	0011
+2	0010
+1	0001
0	0000
−1	1111
−2	1110
−3	1101
−4	1100
−5	1011
−6	1010
−7	1001
$-8 = -2^3$	1000

EXEMPLO 6-3

Qual é a faixa de valores decimais *sem sinal* que pode ser representada com um byte?

Solução

Lembre-se de que um byte são oito bits. Desde que estamos interessados por números sem sinal, não existe bit de sinal e portanto todos os oito bits são usados para magnitude. Portanto, os valores variam desde

$$00000000_2 = 0_{10}$$

até

$$11111111_2 = 255_{10}$$

Isto representa um total de 256 valores diferentes, o que poderíamos prever, pois $2^8 = 256$.

EXEMPLO 6-4

Qual é a faixa de valores decimais *com sinal* que pode ser representada com um byte?

Solução

Tendo em vista que o MSB é usado como bit de sinal, existem sete bits para a magnitude. O maior valor negativo é

$$10000000_2 = -2^7 = -128_{10}$$

O maior valor positivo é

$$01111111_2 = +2^7 - 1 = +127_{10}$$

Logo, a faixa é de −128 até +127, perfazendo um total de 256 valores diferentes, incluindo o zero. De outro modo, já que existem sete bits de magnitude ($N = 7$), então existem $2^{N+1} = 2^8 = 256$ valores diferentes.

EXEMPLO 6-5

Um certo computador está armazenando os dois números com sinal a seguir na sua memória utilizando o sistema de complemento a 2:

$$00011111_2 = +31_{10}$$
$$11110100_2 = -12_{10}$$

Durante a execução de um programa, o computador é instruído a converter cada número para o sinal oposto, isto é, transformar +31 para −31 e −12 para +12. Como ele fará isto?

Solução

Isto é simplesmente a operação de negação, em que um número com sinal pode ter sua polaridade alterada reali-

zando-se apenas a operação de complemento a 2 no número *completo*, incluindo o bit de sinal. Os circuitos do computador lerão o número com sinal da memória, calcularão seu complemento a 2 e retornarão o valor calculado de volta na memória.

Questões de Revisão

1. Represente cada um dos valores a seguir como um número de oito bits com sinal no sistema de complemento a 2.

 (a) +13 **(b)** −7 **(c)** −128
2. Cada um dos itens a seguir apresenta um número binário com sinal no sistema de complemento a 2. Determine o equivalente decimal para cada um deles.

 (a) 100011 **(b)** 1000000 **(c)** 01111110
3. Qual é a faixa de valores decimais com sinal que pode ser representada com 12 bits (incluindo o bit de sinal)?
4. Quantos bits são necessários para representar valores decimais desde −50 até +50?
5. Qual é o maior valor decimal negativo que pode ser representado por um número de dois bytes?
6. Realize a operação de complemento a 2 em cada um dos seguintes itens.

 (a) 10000 **(b)** 10000000 **(c)** 1000
7. Defina a operação de negação.

6-3 ADIÇÃO NO SISTEMA DE COMPLEMENTO A 2

Estudaremos agora como as operações de adição e subtração são realizadas em sistemas digitais que utilizam o complemento a 2 para representar números negativos. Nos vários casos que serão considerados, é importante perceber que a operação realizada sobre os bits de magnitude também é feita sobre o bit de sinal.

Caso I: Dois Números Positivos. A adição de dois números positivos é bastante direta. Considere a adição de +9 e +4:

```
+9 →  | 0 |  1001 (1ª parcela)
+4 →  | 0 |  0100 (2ª parcela)
------------------
      | 0 |  1101 (soma = +13)
        ↑
        └ bits de sinal
```

Observe que os bits de sinal da **primeira parcela** e da **segunda parcela** são ambos iguais a 0, e o bit de sinal do resultado (da soma) é 0, indicando que o número é positivo. Observe também que a primeira e a segunda parcela são escritas de modo a terem o mesmo número de bits. Isto *sempre* deve ser feito em um sistema de complemento a 2.

Caso II: Um Número Positivo e um Outro Menor e Negativo. Considere a adição de +9 e −4. Lembre-se de que −4 estará representado em complemento a 2. Logo, +4 (00100) deve ser convertido para −4 (11100).

```
          ┌ bits de sinal
          ↓
+9 →    | 0 |  1001 (1ª parcela)
−4 →    | 1 |  1100 (2ª parcela)
--------------------
      1̸ | 0 |  0101
      ↑
      └ Este carry é desconsiderado. O resultado é
        00101 (soma = +5).
```

Neste caso, o bit de sinal da segunda parcela é 1. Observe que a operação de adição também é feita sobre os bits de sinal. Na verdade, o carry ("vai-um") é gerado na última posição da adição. *Este carry é sempre descartado*, de modo que a soma final é 00101, que é equivalente a +5.

Caso III: Um Número Positivo e um Outro Maior e Negativo. Considere a adição de −9 e +4:

```
−9 →   10111
+4 →   00100
------------
       11011 (soma = −5)
       ↑
       └── bit de sinal negativo
```

A soma possui o bit de sinal igual a 1, indicando um número negativo. Como o resultado da soma é negativo, ele está representado em complemento a 2, de modo que os últimos quatro bits, 1011, de fato representam o complemento a 2 do resultado. Para encontrar a magnitude verdadeira do resultado, devemos fazer o complemento a 2 de 11011. O resultado é 00101 = +5. Logo, 11011 representa −5.

Caso IV: Dois Números Negativos

```
−9 →    10111
−4 →    11100
-------------
      1̸ 10011
      ↑ ↑
      | └ bit de sinal
      └── Este carry é desconsiderado; o resultado é
          10011 (soma = −13).
```

O resultado final é novamente negativo e está representado em complemento a 2 com o bit de sinal igual a 1. Fazendo o complemento a 2 do resultado temos 01101 = +13.

Caso V: Dois Números Iguais em Magnitude Mas de Sinais Contrários

```
−9 →    10111
+9 →    01001
-------------
0     1̸ 00000
      ↑
      └──── Desconsidere; o resultado é 00000
            (soma = +0).
```

O resultado é obviamente +0, como esperado.

Questões de Revisão

Considere um sistema de complemento a 2 para ambas as questões.

1. *Verdadeiro* ou *falso*: Sempre que a soma de dois números binários com sinal tiver no resultado o bit de sinal igual a 1, a magnitude do resultado estará em complemento a 2.
2. Some os seguintes pares de números com sinal. Escreva o resultado da soma como um número binário com sinal e como um número decimal.

   (a) 100111 + 111011 **(b)** 100111 + 011001

6-4 SUBTRAÇÃO NO SISTEMA DE COMPLEMENTO A 2

A operação de subtração usando o sistema de complemento a 2, na verdade, envolve uma operação de adição e realmente não é diferente do que ocorre nos casos considerados na Seção 6-3 para a operação de adição. Quando subtraímos um número binário (o **subtraendo**) de outro número binário (o **minuendo**), usamos o seguinte procedimento:

1. **Negue o subtraendo.** Isto mudará o subtraendo para um número de igual magnitude mas de sinal oposto.
2. **Adicione o número obtido no passo 1 ao minuendo.** O resultado desta operação de adição representará a *diferença* entre o subtraendo e o minuendo.

Mais uma vez, como em todas as operações em complemento a 2, é necessário que os dois números estejam escritos com o mesmo número de bits.

Vamos considerar o caso em que +4 deve ser subtraído de +9.

minuendo (+9) → 01001
subtraendo (+4) → 00100

Faça a negação do subtraendo para produzir 11100, que representa −4. Agora adicione este valor ao minuendo.

```
  01001    (+9)
+ 11100    (−4)
  -----
1 00101    (+5)
↑
└ Desconsidere; o resultado é 00101 = +5.
```

Quando o subtraendo é representado em complemento a 2, ele, na verdade, torna-se igual a −4. Assim, estamos *adicionando* −4 e +9, que é o mesmo que subtrair +4 de +9. Esta é a mesma situação apresentada no caso II da Seção 6-3. Qualquer operação de subtração, então, torna-se uma operação de adição quando utilizamos o complemento a 2. Esta característica, presente no sistema de complemento a 2, fez com que ele se tornasse um dos sistemas mais utilizados dentre os métodos disponíveis, uma vez que adição e subtração podem ser realizadas pelo mesmo circuito.

O leitor deve verificar os resultados, quando o procedimento descrito anteriormente é aplicado para as seguintes subtrações: (a) +9 − (−4); (b) −9 − (+4); (c) −9 − (−4); (d) +4 − (−4). Lembre-se de que, quando um resultado tem o bit de sinal igual a 1, ele é negativo e está representado em complemento a 2.

Overflow Aritmético

Nos exemplos anteriores de subtração e adição, os números que estavam sendo adicionados eram constituídos de um bit de sinal e quatro bits de magnitude. Qualquer carry para a sexta posição era descartado. Em todos os casos considerados, a magnitude da resposta era pequena o suficiente para ser representada com quatro bits. Vamos ver o que acontece na adição de +9 e +8.

```
+9 →  0  1001
+8 →  0  1000
      -------
      1  0001
```

sinal incorreto ↑ ↑ magnitude incorreta

A resposta tem um bit de sinal negativo, o que está obviamente incorreto, uma vez que estamos adicionando dois números positivos. A resposta correta é +17, mas a magnitude 17 necessita de mais de quatro bits para ser representada, e portanto ocorreu um *overflow* na posição do bit de sinal. Esta condição de **overflow** pode ocorrer apenas quando dois números positivos ou dois números negativos estão sendo somados, e ela sempre produz um resultado incorreto. A ocorrência do overflow pode ser detectada examinando o bit de sinal do resultado e comparando-o com os bits de sinal dos números que estão sendo adicionados. Em um computador, um circuito especial é usado para detectar qualquer condição de overflow para indicar que a resposta está errada. Encontraremos este circuito em um dos exercícios no fim do capítulo.

Questões de Revisão

1. Faça a subtração entre os seguintes pares de números com sinal usando o sistema de complemento a 2. Escreva os resultados como números binários com sinal e como valores decimais.

   (a) 01001 − 11010 **(b)** 10010 − 10011
2. Como um overflow pode ser detectado quando números com sinal estão sendo adicionados?

6-5 MULTIPLICAÇÃO DE NÚMEROS BINÁRIOS

A multiplicação de números binários é feita do mesmo modo que a multiplicação de números decimais. O procedimento, na verdade, é mais simples, uma vez que os dígitos multiplicadores podem ser apenas 0 ou 1, e portanto estaremos multiplicando sempre por 0 e 1 e por nenhum outro dígito. O exemplo seguinte ilustra este procedimento para números binários sem sinal

```
   1001    ←multiplicando = 9₁₀
   1011    ←multiplicando = 11₁₀
   ----
   1001 ┐
  1001  │  produtos parciais
 0000   │
1001    ┘
-------
1100011 }  produto final = 99₁₀
```

Neste exemplo, o multiplicando e o multiplicador estão na forma binária e bits de sinal não estão sendo usados. Os passos seguidos são os mesmos daqueles utilizados na multiplicação decimal. Primeiro, o LSB do multiplicador é examinado. Neste exemplo, ele é igual a 1. Este 1 multiplica o multiplicando produzindo 1001, que é escrito como o primeiro produto parcial. A seguir, o segundo bit do multiplicador é examinado. Ele é igual a 1, e portanto 1001 é escrito como segundo produto parcial. Observe que o segundo produto parcial é *deslocado* de uma posição para a esquerda em relação ao primeiro produto. O terceiro bit do multiplicador é 0, e portanto 0000 é escrito como terceiro produto parcial; novamente, ele é deslocado de uma posição para a esquerda em relação ao produto parcial anterior. O quarto bit do multiplicador é 1, e portanto o último produto parcial é 1001, mais uma vez deslocado de uma posição para a esquerda. Os quatro produtos parciais são então somados para obter o produto final.

A maioria das máquinas digitais pode adicionar apenas dois números binários de cada vez. Por esta razão, os produtos parciais obtidos durante a multiplicação não podem ser somados ao mesmo tempo. Em vez disso, eles são adicionados dois de cada vez, isto é, o primeiro produto parcial é adicionado ao segundo. Este resultado é somado ao terceiro e assim por diante. Este processo é agora ilustrado para o exemplo anterior:

```
Soma {    1001   ← primeiro produto parcial
         1001    ← segundo produto parcial deslocado para a
         ----      esquerda
Soma {   11011   ← soma dos dois primeiros produtos parciais
        0000     ← terceiro produto parcial deslocado para a
        -----      esquerda
Soma {  011011   ← soma dos três primeiros produtos parciais
       1001      ← quarto produto parcial deslocado para a
       ------      esquerda
       1100011   ← soma dos quatro produtos parciais que
                   é igual ao produto total final
                        Soma   Soma
```

Multiplicação no Sistema de Complemento a Dois

Em computadores que usam representação em complemento a dois, a multiplicação é executada do mesmo modo descrito anteriormente, desde que tanto o multiplicador quanto o multiplicando estejam na sua forma binária verdadeira. Se os dois números a serem multiplicados são positivos, eles já estão na sua forma binária verdadeira e são multiplicados nesta forma. O resultado será, obviamente, positivo e o bit de sinal será igual a 0. Quando os números são negativos, eles estarão em complemento a 2. O complemento a 2 de cada um dos números é feito para obtermos números positivos, que serão então multiplicados. O produto é mantido como um número positivo e o bit de sinal é igual a 0.

Quando um dos números é positivo e o outro é negativo, o número negativo é primeiro convertido para um número positivo através do complemento a 2. O resultado obtido representa a verdadeira magnitude do produto. O produto deve ser negativo, uma vez que os números originais possuíam sinais opostos. Logo, ao resultado obtido deve ser aplicado o complemento a 2 e o bit de sinal deve ser igual a 1.

Questões de Revisão

1. Multiplique os números sem sinal 0111 e 1110.

6-6 DIVISÃO BINÁRIA

O processo para dividir um número binário (o *dividendo*) por um outro (o *divisor*) é o mesmo que é utilizado para números decimais, e ao qual geralmente nos referimos como "divisão longa". Na verdade, o processo é mais simples em binário, porque quando estamos verificando quantas vezes o divisor "cabe" no dividendo existem apenas duas possibilidades, 0 ou 1. Para ilustrar, considere os dois exemplos simples de divisão binária:

```
1001 | 11        1010,0 | 100
011  |----       100    |-----
     | 11               | 10,1
----             ------
0011             0010,0
  11                100
----             ------
   0                  0
```

No primeiro exemplo, temos 1001_2 dividido por 11_2, que é equivalente a $9 \div 3$ em decimal. O quociente resultante é $0011_2 = 3_{10}$. No segundo exemplo, 1010_2 é dividido por 100_2, ou $10 \div 4$ em decimal. O resultado é $0010,1_2 = 2,5_{10}$.

Na maioria dos sistemas digitais, as subtrações que são parte da operação de divisão são geralmente executadas usando complemento a 2, isto é, tomando o complemento a 2 do subtraendo para depois adicioná-lo.

A divisão de números com sinal é tratada do mesmo modo que na multiplicação. Números negativos são transformados em números positivos por complemento, para que a divisão seja executada posteriormente. Se o dividendo e o divisor tiverem sinais opostos, o quociente resultante é transformado em um número negativo tomando o seu complemento a 2 e colocando o bit de sinal em 1. Se o dividendo e o divisor tiverem o mesmo sinal, o quociente é deixado como um número positivo e o bit de sinal é colocado em 0.

6-7 ADIÇÃO BCD

No Cap. 2 afirmamos que muitos computadores e calculadoras usam o código BCD para representar números decimais. Lembre-se de que este código toma *cada* dígito decimal e o representa como um código de 4 bits que vai de 0000 a 1001. A adição de dois números decimais que estão representados em BCD pode ser mais bem compreendida considerando os dois casos que podem ocorrer quando dois dígitos decimais são adicionados.

Soma Menor ou Igual a 9

Considere a adição de 5 e 4 usando BCD para representar cada dígito:

5	0101	← BCD para 5
+4	+ 0100	← BCD para 4
9	1001	← BCD para 9

A adição é executada como uma adição binária normal, e o resultado é 1001, que é o código BCD para 9. Como um outro exemplo, veja a adição de 45 e 33:

45	0100	0101	← BCD para 45
+33	+ 0011	0011	← BCD para 33
78	0111	1000	← BCD para 78

Neste exemplo, os códigos de 4 bits de 5 e 3 são somados em binário para produzir 1000, que é o código BCD para 8. De modo similar, a adição dos dígitos mais significativos produz 0111, que é o código BCD para 7. O total é 01111000, que é o código BCD para 78.

Nos exemplos anteriores, nenhuma das somas dos pares de dígitos decimais excedeu 9, portanto, *nenhum vai-um decimal foi produzido*. Para estes casos, o processo de adição em BCD é bastante direto, e na verdade é idêntico à adição binária.

Soma Maior do que 9

Considere a adição de 6 e 7 em BCD:

6	0110	← BCD para 6
+7	+ 0111	← BCD para 7
+13	1101	← código BCD inválido

O resultado da soma 1101 não existe no código BCD. Ele é um dos seis códigos proibidos ou inválidos. Isto aconteceu porque a soma dos dois dígitos excede 9. Sempre que isto ocorrer, o resultado da soma deve ser corrigido através da adição do número 6 (0110), para que não haja códigos inválidos no resultado:

	0110	← BCD para 6
	+ 0111	← BCD para 7
	1101	← soma inválida
	0110	← soma 6 para corrigir
0001	0011	← BCD para 13
1	3	

Como foi mostrado anteriormente, 0110 é adicionado à soma inválida para produzir o resultado correto em BCD. Observe que um carry é produzido para a segunda posição decimal. Esta adição de 0110 deve ser realizada sempre que a soma dos dois dígitos decimais for maior que 9.

Como um outro exemplo, veja a soma de 47 e 35 em BCD:

47	0100	0111	← BCD para 47
+35	+ 0011	0101	← BCD para 35
82	0111	1100	← soma inválida no primeiro dígito
	1 ←	0110	← soma 6 para corrigir
	1000	0010	← soma BCD correta
	8	2	

A adição dos códigos de 4 bits para os dígitos 7 e 5 resulta em uma soma inválida que deve ser corrigida através da adição de 0110. Observe que isto gera um carry, que deve ser adicionado à soma BCD dos dígitos da segunda posição.

Considere a adição de 59 e 38 em BCD:

	1 ←		
59	0101	1001	← BCD para 59
+38	+ 0011	1000	← BCD para 38
97	1001	0001	← realiza a adição
		0110	← soma 6 para corrigir
	1001	0111	BCD para 97
	9	7	

Neste caso, a adição dos dígitos menos significativos (LSDs) produz 17 = 10001 como resultado. Isto gera um carry para a próxima posição que deve ser adicionado aos códigos para 5 e 3. Uma vez que 17 > 9, um fator de correção 6 deve ser adicionado ao LSD da soma. A adição deste fator não gera um carry, pois este já foi gerado na adição original.

Resumindo o procedimento de adição BCD:

1. Usando a adição binária comum, some os grupos de código BCD para cada posição de dígito.
2. Para as posições onde a soma é menor ou igual a 9, nenhuma correção é necessária. O resultado está na forma BCD apropriada.
3. Quando a soma dos dois dígitos é maior que 9, o fator de correção 0110 deve ser adicionado para obter o resultado na forma BCD correta. Este caso sempre produz um carry para a próxima posição, seja da adição original (passo 1), seja da adição de correção.

O procedimento para a adição em BCD é, obviamente, mais complicado do que a adição binária direta. Isto também é verdadeiro para as outras operações aritméticas em BCD. Os leitores devem realizar a adição de 275 + 641, e depois comparar com o procedimento correto mostrado a seguir.

275	0010	0111	0101	←BCD para 275
+641	+ 0110	0100	0001	←BCD para 641
916	1000	1011	0110	←realiza a adição
	+	0110		←soma 6 para corrigir
	1001	0001	0110	←o segundo dígito BCD para 916

Questões de Revisão

1. Como você pode dizer que uma correção é necessária na adição BCD?
2. Represente 135_{10} e 265_{10} em BCD e depois realize a adição BCD. Verifique se o resultado está correto, convertendo-o novamente para decimal.

6-8 ARITMÉTICA HEXADECIMAL

Números hexadecimais são amplamente utilizados na programação de computadores em linguagem de máquina e

na especificação de endereços de memória de computadores. Quando estivermos trabalhando nestas áreas, você encontrará situações em que números hexa devem ser adicionados ou subtraídos.

Adição em Hexa

A adição de números hexadecimais é feita basicamente do mesmo modo que a adição decimal desde que você se lembre de que o maior dígito hexa é F em vez de 9. O seguinte procedimento é sugerido:

1. Some os dois dígitos hexadecimais em decimal, inserindo mentalmente o decimal equivalente para os dígitos maiores do que 9.
2. Se a soma é menor ou igual a 15, ele pode ser expresso por um dígito hexadecimal.
3. Se a soma é maior ou igual a 16, subtraia 16 e coloque um carry na próxima posição.

Os exemplos a seguir vão ilustrar este procedimento.

EXEMPLO 6-6

Some os números hexadecimais 58 e 24.

Solução

```
 58
+24
 7C
```

A soma dos LSDs (8 e 4) produz 12, que é C em hexa. Não existe carry para a próxima posição. A soma de 5 com 2 é igual a 7.

EXEMPLO 6-7

Some os números hexadecimais 58 e 4B.

Solução

```
 58
+4B
 A3
```

Comece pela adição de 8 e B, substituindo mentalmente B por 11. Isto produz como resultado 19. Uma vez que 19 é maior do que 16, subtraia 16 para obter 3 como resultado. Anote o 3 e coloque um carry na próxima posição. Este carry é somado a 5 e 4 para produzir 10_{10} como resultado, que deve ser convertido para o hexadecimal A.

EXEMPLO 6-8

Some 3AF e 23C.

Solução

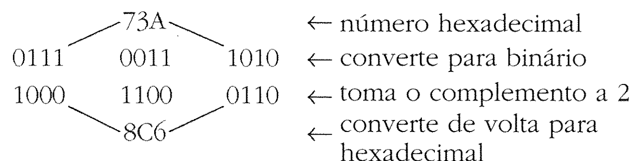
```
 3AF
+23C
 5EB
```

A soma de F e C é considerada como se fosse 15 + 12 = 27_{10}. Uma vez que o resultado é maior que 16, subtraia 16 para obter 11_{10}, que é o hexadecimal B, e coloque 1 na segunda posição. Some este carry com A e 3 para obter E. Não existe carry da posição MSD.

Subtração Hexadecimal

Lembre-se de que números hexadecimais são apenas um modo eficiente de representar números binários. Então podemos subtrair números hexadecimais usando o mesmo método utilizado para números binários. O complemento a 2 do subtraendo hexa deverá ser tomado e depois *adicionado* ao minuendo, e qualquer carry da posição MSD deverá ser descartado.

Como fazemos para encontrar o complemento a dois de um número hexadecimal? Um modo é convertê-lo para binário, tomar o complemento a dois do binário equivalente e depois convertê-lo de volta a hexadecimal. O processo é ilustrado a seguir.



Existe um procedimento mais rápido: subtraia *cada* dígito hexadecimal de F e depois adicione 1. Vamos experimentar este procedimento para o mesmo número hexadecimal do exemplo anterior.

```
 F   F   F
-7  -3  -A   } ← subtrai cada dígito de F
 8   C   5
        +1   ← soma 1
 8   C   6   ← equivalente hexadecimal do complemento a 2
```

Experimente um dos procedimentos anteriores no número hexadecimal E63. O resultado correto para o complemento a 2 é 19D.

EXEMPLO 6-9

Subtraia $3A5_{16}$ de 592_{16}.

Solução

Primeiro, converta o subtraendo (3A5) para o seu complemento a 2 utilizando um dos métodos apresentados anteriormente. O resultado é C5B. A seguir adicione este valor ao minuendo (592):

```
  592
+ C5B
 1̸1ED
 ↑—— Desconsidere o carry
```

Ignorando o carry gerado na adição dos MSDs, o resultado é 1ED. Podemos provar que ele está correto adicionando 1ED a 3A5 e verificando se ele é igual a 592_{16}.

Representação Hexadecimal de Números com Sinal

Os dados armazenados na memória interna de um microcomputador ou em um disco rígido, ou em CD-ROM, são geralmente armazenados em bytes (grupos de oito bits). O byte armazenado em um lugar particular na memória é, geralmente, expresso em hexadecimal, pois é mais eficiente e menos sujeito a erros do que expressá-lo em binário. Quando os dados consistem em números *com sinal*, é bastante útil ser capaz de reconhecer se um valor hexadecimal representa um número positivo ou negativo. Por exemplo, a Tabela 6-2 relaciona os bytes armazenados em um pequeno segmento de memória começando no endereço 4000.

TABELA 2-1

Endereço Hexa	Dados Binários Armazenados	Valor Hexa	Valor Decimal
4000	00111010	3A	+58
4001	11100101	E5	−29
4002	01010111	57	+87
4003	10000000	80	−128

Em cada posição de memória, o dado, que na verdade representa um número decimal com sinal, é armazenado como um byte. O equivalente hexadecimal deste valor binário é um número hexadecimal de dois dígitos. Sempre que o valor do dado binário é negativo, o bit de sinal (MSB) é igual a 1. Isto faz com que o MSD do número hexadecimal seja maior ou igual a 8 (isto é, 8, 9, A, B, C, D, E ou F). Quando o valor do dado binário é positivo (0 no bit de sinal), o MSD do número hexadecimal é sempre menor ou igual a 7 (isto é, 7, 6, 5, 4, 3, 2, 1 ou 0). Esta característica faz com que seja fácil determinar se um número hexadecimal representa um número positivo ou negativo.

Questões de Revisão

1. Some 67F + 2A4.
2. Subtraia 67F − 2A4.
3. Quais dos seguintes números hexadecimais representam números positivos: 2F, 77EC, C000, 6D, FFFF?

6-9 CIRCUITOS ARITMÉTICOS

Uma função essencial da maioria dos computadores e calculadoras é a realização de operações aritméticas. Estas operações são todas realizadas na unidade lógica e aritmética de um computador, onde portas lógicas e flip-flops são combinados de tal modo que eles podem somar, subtrair, multiplicar e dividir números binários. Estes circuitos realizam operações aritméticas em velocidades que não são humanamente possíveis. Geralmente, uma operação de adição demora menos de 100 ns.

Estudaremos agora alguns dos circuitos aritméticos básicos que são usados para realizar as operações aritméticas discutidas anteriormente. Em alguns casos, iremos através do processo de projeto, mesmo que circuitos possam estar comercialmente disponíveis na forma de circuito integrado, para exercitar as técnicas aprendidas no Cap. 4.

Unidade Lógica e Aritmética

Todas as operações aritméticas são realizadas na **unidade lógica e aritmética (ULA)** de um computador. A Fig. 6-3 apresenta um diagrama de blocos mostrando os elementos principais em uma ULA. O objetivo principal de uma ULA é aceitar dados binários armazenados na memória e executar as operações lógicas e aritméticas sobre estes dados de acordo com as instruções da unidade de controle.

A unidade lógica e aritmética contém pelo menos dois registradores: o *registrador B* e o **registrador acumulador**. Ela também possui lógica combinacional, que realiza as operações lógicas e aritméticas sobre os números binários que estão armazenados no registrador *B* e no acumulador. Uma seqüência típica pode ocorrer como se segue:

1. A unidade de controle recebe uma instrução (da unidade de memória) especificando que um número armazenado em uma posição particular de memória (endereço) deve ser adicionado ao número que está atualmente armazenado no acumulador.
2. O número a ser adicionado é transferido da memória para o registrador *B*.
3. O número no registrador *B* e o número no acumulador são somados nos circuitos lógicos (ao comando da unidade de controle). O resultado é então enviado ao acumulador para ser armazenado.
4. O novo número no acumulador pode aí permanecer para que um outro número possa ser somado a ele, ou, se o processo aritmético em particular está terminado, ele pode ser transferido para a memória.

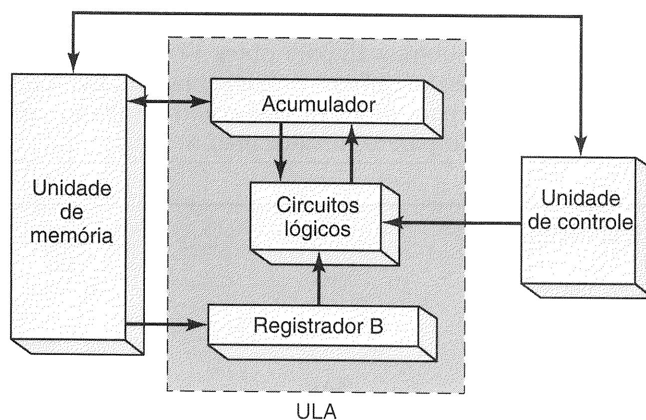


Fig. 6-3 - Blocos funcionais de uma ULA.

Estes passos devem deixar claro como o acumulador obteve seu nome. Este registrador "acumula" os resultados que ocorrem quando realizamos adições sucessivas entre os novos números adquiridos da memória e os resultados previamente acumulados. Na verdade, para qualquer problema aritmético que contenha vários passos, o acumulador geralmente contém os resultados de passos intermediários, à medida que eles vão sendo executados, bem como o resultado final quando o problema está terminado.

6-10 SOMADOR BINÁRIO PARALELO

Computadores e calculadoras realizam a operação de adição sobre dois números binários de cada vez, onde cada número binário pode ter vários dígitos binários. A Fig. 6-4 ilustra a adição de dois números de cinco bits. A 1.ª parcela é armazenada no registrador acumulador, isto é, o acumulador contém cinco flip-flops que armazenam os valores 10101 em flip-flops sucessivos. De modo similar, a 2.ª parcela, o número que deve ser somado à 1.ª, está armazenado no registrador *B*, que neste caso é 00111.

O processo de adição é iniciado pela soma dos bits menos significativos (LSBs) das parcelas. Então, $1 + 1 = 10$, o que significa que a *soma* para esta posição é 0, com *carry* igual a 1.

Este carry é adicionado à próxima posição, juntamente com os bits das parcelas desta posição. O mesmo processo é repetido para as posições remanescentes, como mostra a Fig. 6-4.

Em cada passo desta adição estamos realizando a soma de três bits: o bit da 1.ª parcela, o bit da 2.ª parcela e o bit de carry da posição anterior. O resultado da adição destes três bits produz como resultado dois bits: um bit de *soma* e um bit de *carry* que será adicionado à próxima posição. Deve estar claro que o mesmo procedimento é seguido para cada posição de bit. Portanto, se pudermos projetar um circuito lógico que possa implementar este processo, então tudo o que temos que fazer é usar circuitos idênticos para cada uma das posições de bit. Isto está mostrado na Fig. 6.5

Neste diagrama, as variáveis A_4, A_3, A_2, A_1 e A_0 representam os bits da 1.ª parcela que estão armazenados no acumulador (que também é chamado de registrador *A*). As variáveis B_4, B_3, B_2, B_1 e B_0 representam os bits da 2.ª parcela armazenados no registrador *B*. As variáveis C_4, C_3, C_2, C_1 e C_0 representam os bits de carry das posições correspondentes. As variáveis S_4, S_3, S_2, S_1, S_0 são o resultado da soma dos bits de cada posição. Os bits correspondentes das parcelas são fornecidos como entradas para um circuito lógico chamado **somador completo**, juntamente com o bit de carry da posição anterior. Por exemplo, os bits A_1 e B_1 são as entradas do somador completo 1, juntamente com o bit de carry da posição anterior, que foi produzido pela adição dos bits A_0 e B_0. Os bits A_0 e B_0 são entradas do somador completo 0, juntamente com C_0. Uma vez que A_0 e B_0 são os LSBs das parcelas, poderia parecer que C_0 devesse ser sempre igual a 0, uma vez que não há carry para esta posição. Entretanto, veremos mais tarde que existirão situações nas quais C_0 pode ser igual a 1.

O circuito do somador completo usado em cada uma das posições tem três entradas: um bit *A*, um bit *B* e um bit *C*. Ele também produz duas saídas: um bit de soma e um bit de carry. Por exemplo, o somador completo 0 tem como entradas A_0, B_0 e C_0 e fornece como saídas S_0 e C_1. O somador completo 1 tem A_1, B_1 e C_1 como entradas e S_1 e C_2 como saídas, e assim sucessivamente. Este arranjo pode ser repetido por tantas posições (bits) quantas existam nas parcelas. Embora este exemplo seja para números de cinco bits, nos computadores modernos estes números geralmente possuem de 8 a 64 bits.

O arranjo mostrado na Fig. 6-5 é chamado de **somador paralelo** porque todos os bits das parcelas estão presentes e são apresentados aos circuitos somadores *simultaneamente*. Isto significa que as adições em cada posição acontecem ao mesmo tempo. Esta maneira difere daquela que usamos quando fazemos as adições a mão, pois neste caso tomamos uma posição de cada vez, começando pelo LSB. Obviamente, a adição em paralelo é bastante rápida. Adiante diremos mais sobre isto.

Questões de Revisão

1. Quantas entradas um somador completo possui? Quantas saídas?
2. Considere os seguintes níveis de entrada na Fig. 6-5: $A_4A_3A_2A_1A_0 = 01001$, $B_4B_3B_2B_1B_0 = 00111$ e $C_0 = 0$.
   (a) Quais são os níveis lógicos na saída do somador completo número 2?
   (b) Qual é o nível lógico da saída C_5?

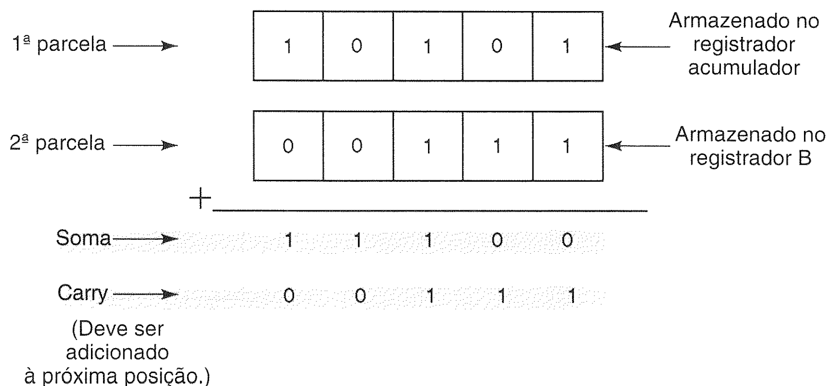


Fig. 6-4 - Processo de adição binária típico.

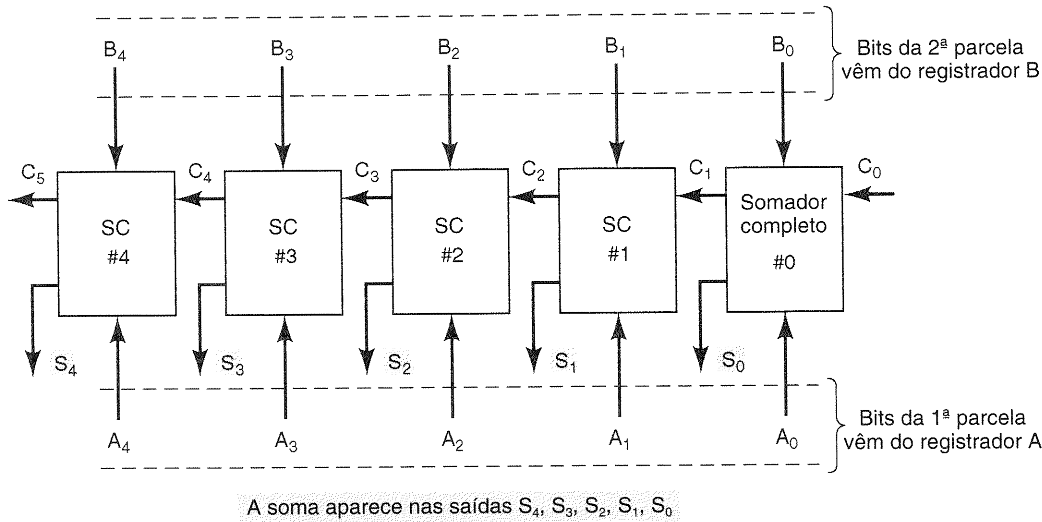


Fig. 6-5 - Diagrama de blocos de um somador paralelo de cinco bits utilizando somadores completos.

6-11 PROJETO DE UM SOMADOR COMPLETO

Agora que já sabemos a função de um somador completo, podemos passar para o projeto de um circuito lógico que vai realizar esta função. Primeiro, vamos construir a tabela-verdade mostrando os valores das saídas para todas as combinações possíveis das entradas. A Fig. 6-6 mostra uma tabela-verdade com três entradas A, B, C_{IN}, e duas saídas S e C_{OUT}. Existem oito combinações possíveis para as entradas, e para cada uma delas estão relacionados os valores desejados das saídas. Por exemplo, considere o caso $A = 1$, $B = 0$ e $C_{IN} = 1$. O somador completo (daqui por diante abreviado **SC**) deve adicionar estes bits para produzir um bit de soma (S) igual a 0 e um bit carry (C_{OUT}) igual a 1. O leitor deve verificar os outros casos para ter certeza de que eles foram compreendidos.

Uma vez que existem duas saídas, projetaremos circuitos individuais para cada saída, começando com a saída S. A tabela-verdade mostra que existem quatro casos para os quais S deve ser igual a 1. Escrevendo a expressão para S na forma de soma-de-produtos, temos

$$S = \bar{A}\bar{B}C_{IN} + \bar{A}B\bar{C}_{IN} + A\bar{B}\bar{C}_{IN} + ABC_{IN} \qquad (6\text{-}1)$$

Podemos agora simplificar esta expressão por meio da fatoração. Infelizmente, nenhum dos termos nesta expressão possui duas variáveis em comum com qualquer um dos outros termos. Entretanto, $\bar{A}$ pode ser fatorado nos primeiros dois termos, e A pode ser fatorado nos dois últimos:

$$S = \bar{A}(\bar{B}C_{IN} + B\bar{C}_{IN}) + A(\bar{B}\bar{C}_{IN} + BC_{IN})$$

O primeiro termo entre parênteses pode ser reconhecido como a função EX-OR de B e C_{IN}, que pode ser escrita como $B \oplus C_{IN}$. O segundo termo entre parênteses corresponde a um EX-NOR de B e C_{IN}, que pode ser escrito como $B \oplus C_{IN}$ Assim, a expressão para S torna-se

$$S = \bar{A}(B \oplus C_{IN}) + A(\overline{B \oplus C_{IN}})$$

Se fizermos $X = B \oplus C_{IN}$, S pode ser escrita como:

Bit de entrada da 1ª parcela	Bit de entrada da 2ª parcela	Bit de entrada do carry	Bit de saída da soma	Bit de saída do carry
A	B	C_{IN}	S	C_{OUT}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

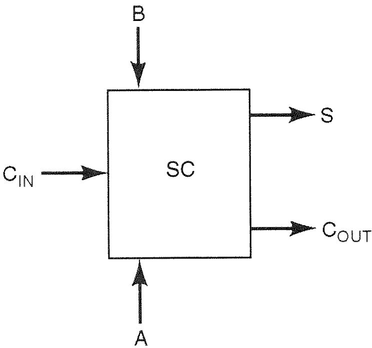


Fig. 6-6 - Tabela-verdade de um circuito somador completo.

$$S = \overline{A} \cdot X + A \cdot \overline{X} = A \oplus X$$

que é simplesmente o EX-OR de A e X. Substituindo a expressão para X, teremos:

$$S = A \oplus [B \oplus C_{IN}] \qquad (6\text{-}2)$$

Considere agora a saída C_{OUT} na tabela-verdade da Fig. 6-6. Podemos escrever uma expressão na forma de soma-de-produtos para C_{OUT} como se segue:

$$C_{OUT} = \overline{A}BC_{IN} + A\overline{B}C_{IN} + AB\overline{C}_{IN} + ABC_{IN}$$

Esta expressão pode ser simplificada por meio de fatoração. Empregaremos o truque que foi introduzido no Cap. 4, onde usaremos o termo ABC_{IN} *três* vezes, uma vez que ele tem fatores comuns com cada um dos outros termos. Assim,

$$C_{OUT} = BC_{IN}(\overline{A} + A) + AC_{IN}(\overline{B} + B) + AB(\overline{C}_{IN} + C_{IN}) =$$
$$= BC_{IN} + AC_{IN} + AB \qquad (6\text{-}3)$$

Esta expressão não pode ser simplificada além disso.

As expressões (6-2) e (6-3) podem ser implementadas como está mostrado na Fig. 6-7. Várias outras implementações podem ser usadas para as expressões de S e C_{OUT}, nenhuma das quais possui qualquer vantagem em particular sobre aquela que foi mostrada. O circuito completo com as entradas A, B, C_{IN} e com as saídas S e C_{OUT} representa o somador completo. Cada um dos SCs na Fig. 6-5 contém este mesmo circuito (ou um outro equivalente).

Simplificação com o Mapa de Karnaugh

Simplificamos as expressões para S e C_{OUT} usando métodos algébricos. O mapa de Karnaugh também pode ser utilizado. A Fig. 6-8 (a) mostra o mapa K para a saída S. Este mapa não possui 1s adjacentes, e portanto não há pares ou quartetos para agrupar. Portanto, a expressão para S não pode ser simplificada usando o mapa K. Isto demonstra uma limitação do método do mapa de Karnaugh quando comparado com o método algébrico. Fomos capazes de simplificar a expressão para S através de fatoração e do uso de operações EX-OR e EX-NOR.

	$\overline{C}_{IN}$	C_{IN}
$\overline{A}\overline{B}$	0	1
$\overline{A}B$	1	0
AB	0	1
$A\overline{B}$	1	0

Mapa K para S

$= \overline{A}\overline{B}C_{IN} + \overline{A}B\overline{C}_{IN} + ABC_{IN} + A\overline{B}\overline{C}_{IN}$

(a)

	$\overline{C}_{IN}$	C_{IN}
$\overline{A}\overline{B}$	0	0
$\overline{A}B$	0	1
AB	1	1
$A\overline{B}$	0	1

Mapa K para C_{OUT}

$C_{OUT} = BC_{IN} + AC_{IN} + A$

(b)

Fig. 6-8 - Mapas de Karnaugh para as saídas do somador completo.

O mapa K para C_{OUT} é mostrado na Fig. 6-8(b). Os três pares que são agrupados vão produzir a mesma expressão obtida utilizando o método algébrico.

Meio Somador

Um somador completo opera sobre três entradas para produzir um bit de soma e um bit de carry. Em alguns casos, é necessário um circuito que some apenas dois bits e que

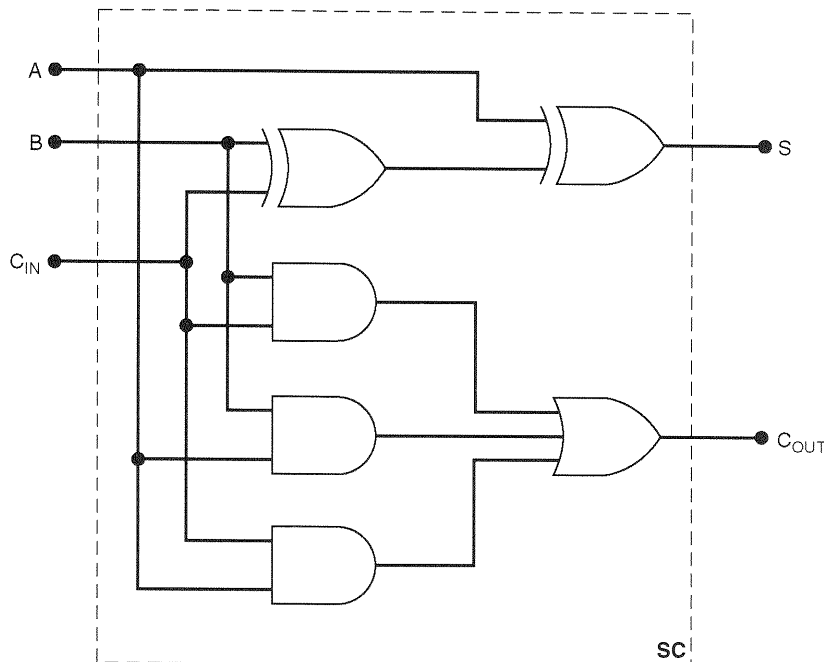
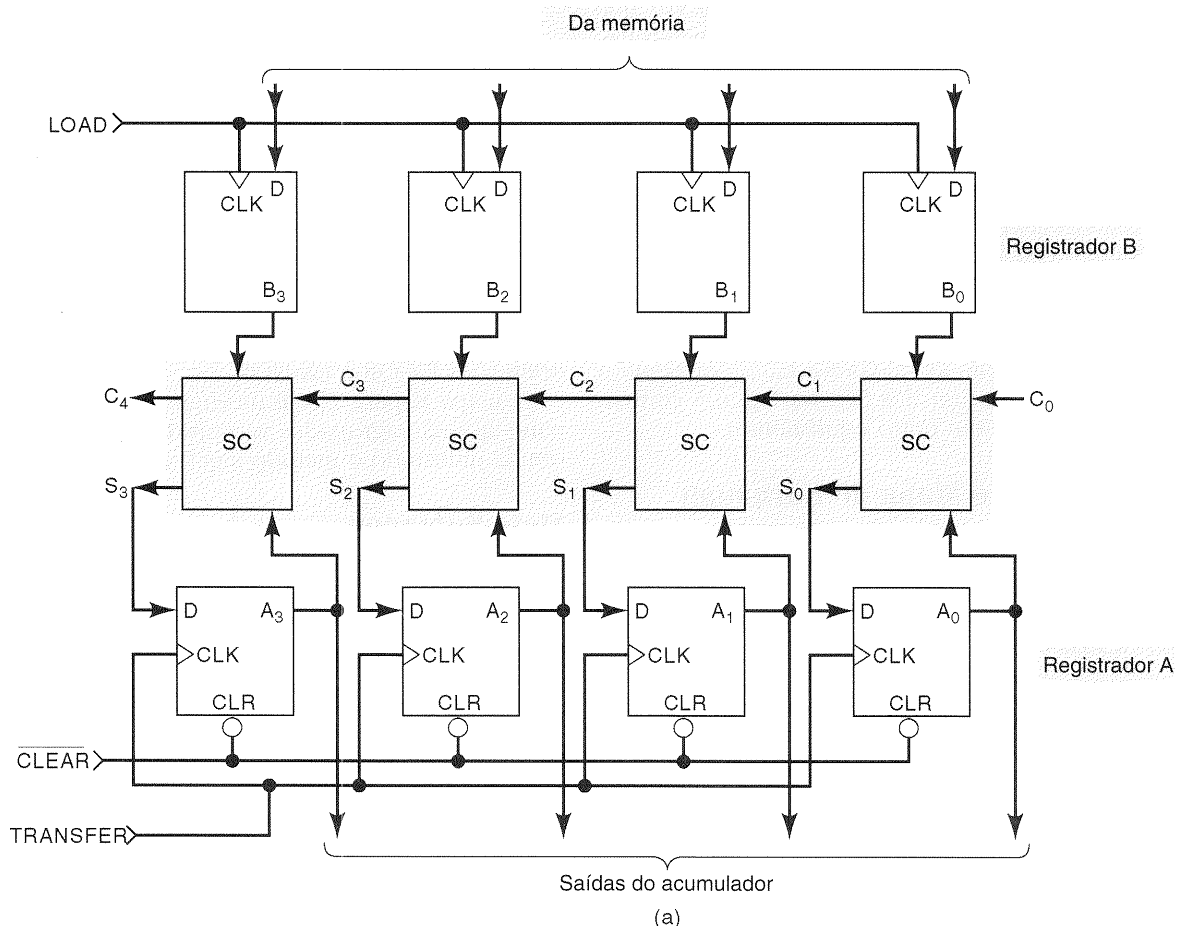


Fig. 6-7 - Circuito para um somador completo.

produza o bit de soma e o bit de carry. Um exemplo seria a adição dos bits menos significativos de dois números binários, onde não há entrada de carry para ser adicionada. Um circuito especial pode ser projetado para receber *dois* bits de entrada, *A* e *B*, e produzir as saídas soma (*S*) e carry (C_{OUT}). Este circuito é chamado de **meio somador (MS)**. Sua operação é similar à do somador completo, exceto pelo fato de que ele opera com apenas dois bits. Deixaremos o projeto do MS como um exercício no final do capítulo (veja Problema 6-19).

6-12 SOMADOR PARALELO COMPLETO COM REGISTRADORES

Em um computador, os números a serem adicionados são armazenados em registradores. A Fig. 6-9 mostra o diagrama de um somador paralelo de quatro bits, incluindo os registradores de armazenamento. Os bits da primeira parcela, A_3 até A_0, estão armazenados no acumulador (registrador *A*). Os bits da segunda parcela, B_3 até B_0, estão armaze-



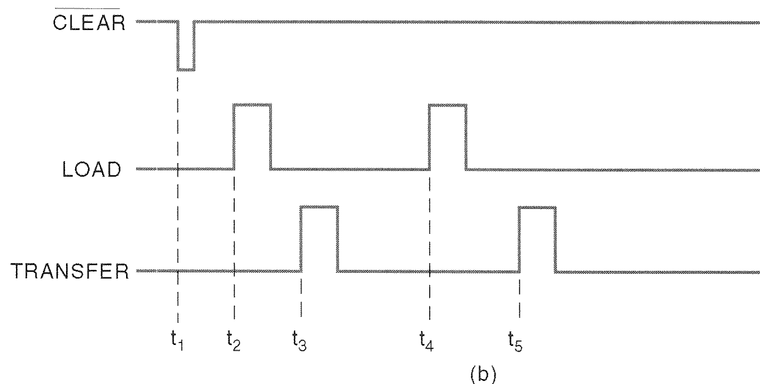


Fig. 6-9 - (a) Somador paralelo de quatro bits completo com registradores. (b) Sinais usados para adicionar números binários da memória e armazenar sua soma no acumulador.

nados no registrador B. Cada um destes registradores é construído a partir de FFs do tipo D para facilitar a transferência de dados.

O conteúdo do registrador A (isto é, o número binário armazenado em A_3 até A_0) é adicionado ao conteúdo do registrador B através de quatro somadores completos, produzindo a soma que aparece nas saídas S_3 a S_0. C_4 é o carry do quarto somador, e pode ser usado como entrada de carry para um quinto somador ou como um bit de *overflow* para indicar que a soma excedeu 1111.

Observe que os bits da soma estão conectados às entradas D do registrador A. Isto permite uma transferência paralela do resultado da soma para o registrador A na transição positiva do pulso TRANSFER. Deste modo, o resultado pode ser armazenado no registrador A.

Observe também que as entradas D do registrador B estão vindo da memória do computador, de modo que números binários podem ser transferidos em paralelo da memória para o registrador B na transição positiva do pulso LOAD. Na maioria dos computadores existe também a previsão de que números binários possam ser transferidos em paralelo da memória para o registrador A. Por simplicidade, o circuito necessário para realizar esta transferência não é mostrado neste diagrama. Ele será visto em um exercício no fim do capítulo.

Finalmente, observe que as saídas do registrador A estão disponíveis para permitir a transferência do conteúdo de A para um outro registrador, ou para a memória do computador. Isto permitirá que o circuito somador esteja disponível para um novo conjunto de números.

Notação para Registradores

Antes de continuar no estudo do processo completo pelo qual este circuito adiciona dois números binários, será bastante útil introduzir uma notação que torne mais fácil a descrição dos conteúdos dos registradores e das operações de transferência de dados.

Sempre que quisermos fornecer os níveis presentes em cada FF de um registrador ou de cada saída de um grupo de saídas, usaremos colchetes, conforme ilustrado a seguir

$$[A] = 1011$$

Isto é o mesmo que escrever $A_3 = 1$, $A_2 = 0$, $A_1 = 1$, $A_0 = 1$. Em outras palavras pense em $[A]$ como uma representação do "conteúdo de A".

Sempre que quisermos indicar transferência de dados de um registrador de/para um registrador, utilizaremos uma seta, conforme indicado a seguir:

$$[B] \rightarrow [A]$$

Isto significa que o conteúdo do registrador B foi transferido para o registrador A. O conteúdo anterior do registrador A foi perdido como resultado desta operação, e o registrador B permanecerá inalterado.

Seqüência de Operações

Descreveremos agora o processo pelo qual o circuito da Fig. 6-9 irá adicionar os números 1001 e 0101. Considere que $C_0 = 0$, isto é, não há carry para a posição LSB.

1. **$[A] = 0000$**. Um pulso CLEAR é aplicado às entradas assíncronas (*CLR*) de cada FF no registrador A. Isto ocorre no instante t_1.
2. **$[M] \rightarrow [B]$**. Este primeiro número binário é transferido da memória (M) para o registrador B. Neste caso, o número binário 1001 é carregado no registrador B na transição positiva do pulso LOAD em t_2.
3. **$[S]^* \rightarrow [A]$**. Com $[B] = 1001$ e $[A] = 0000$, os somadores completos produzem o resultado 1001, isto é, $[S] = 1001$. Estes bits de saída do resultado são transferidos para o registrador A na transição positiva do pulso TRANSFER em t_3. Isto faz com que $[A] = 1001$.
4. **$[M] \rightarrow [B]$**. O segundo número binário, 0101, é transferido da memória para o registrador B na transição positiva do segundo pulso LOAD em t_4. Isto faz com que $[B] = 0101$.
5. **$[S] \rightarrow [A]$**. Com $[B] = 0101$ e $[A] = 1001$, os somadores completos produzem $[S] = 1110$. Estes bits de saída da soma são transferidos para o registrador A quando o segundo pulso TRANSFER ocorre em t_5. Então, $[A] = 1110$.
6. Neste ponto, a soma dos dois números binários está no acumulador, $[A]$, sendo geralmente transferida para a memória do computador, permitindo que o circuito somador possa ser usado por um novo conjunto de números. O circuito que realiza $[A] \rightarrow [M]$ não está mostrado na Fig. 6-9.

Questões de Revisão

1. Suponha que quatro diferentes números de quatro bits são tomados da memória e somados pelo circuito da Fig. 6-9. Quantos pulsos CLEAR serão necessários? Quantos pulsos TRANSFER? Quantos pulsos LOAD?
2. Determine o conteúdo do registrador A depois da seguinte seqüência de operações: $[A] = 0000$, $[0110] \rightarrow [B]$, $[S] \rightarrow [A]$, $[1110] \rightarrow [B]$, $[S] \rightarrow [A]$.

6-13 PROPAGAÇÃO DO CARRY

O somador paralelo da Fig. 6-9 realiza adições em uma velocidade relativamente alta, uma vez que ele soma os bits de cada posição simultaneamente. Entretanto, sua velocidade é limitada por um efeito chamado **propagação do carry**, que pode ser mais bem explicado considerando a seguinte adição:

$$\begin{array}{r} 0111 \\ +\ 0001 \\ \hline 1000 \end{array}$$

A adição dos bits menos significativos produz um carry para a segunda posição. Este carry, quando somado aos bits da segunda posição, gera um carry para a terceira posição. Este último carry, quando adicionado aos bits da terceira posição, produz um carry para a última posição. A coisa mais importante a ser observada neste exemplo é que o bit de sinal gerado na *última* posição (MSB) depende do carry que foi gerado na *primeira* posição (LSB).

*Apesar de S não ser um registrador, usaremos $[S]$ para representar o grupo de saídas S.

Olhando o circuito da Fig. 6-9 por este ponto de vista, o bit S_3 do último somador completo depende do bit C_1 do primeiro somador completo. Entretanto, o sinal C_1 deve passar por três somadores intermediários antes que possa produzir S_3. Isto significa que a saída S_3 não vai atingir seu valor correto até que C_1 se propague pelos somadores intermediários. Isto representa um atraso que depende do atraso de propagação do circuito somador completo. Por exemplo, se cada somador tem um atraso de propagação de 40 ns, S_3 só atingirá seu nível correto 120 ns após C_1 ter sido gerado. Isto significa que o pulso que comanda a soma só pode ser aplicado 160 ns após as parcelas terem sido carregadas nos seus registradores (os 40 ns extras são devidos ao atraso do somador completo do LSB, que gera C_1).

Obviamente, esta situação torna-se muito pior se ampliarmos o circuito somador para adicionar um número maior de bits. Se o somador estiver manipulando números de 32 bits, o atraso de propagação do carry seria de 1280 ns = 1,28 μs. O comando de soma só poderia ser aplicado após, pelo menos, 1,28 μs de os números estarem presentes nos registradores.

Esta ordem de grandeza do atraso de propagação é proibitiva para computadores de alta velocidade. Felizmente, os projetistas de circuitos lógicos desenvolveram uma série de esquemas engenhosos para reduzir este atraso. Um destes esquemas, chamado de **carry antecipado**, utiliza portas lógicas para observar os bits de mais baixa ordem das parcelas para ver se um carry de mais alta ordem deve ser gerado. Por exemplo, é possível construir um circuito lógico com B_2, B_1, B_0, A_2, A_1 e A_0 como entradas e C_3 como saída. Este circuito lógico deve ter um atraso menor do que aquele resultante da propagação do carry pelos somadores. Este esquema requer uma grande quantidade de circuitos adicionais, mas isto é necessário para se obter somadores de alta velocidade. Estes circuitos adicionais não representam um limite significativo devido ao uso de circuitos integrados. Muitos somadores de alta velocidade disponíveis na forma de circuito integrado utilizam o carry antecipado ou uma outra técnica similar para reduzir o atraso de propagação total.

6-14 SOMADOR PARALELO INTEGRADO

Diversos somadores paralelos estão disponíveis como CIs. O mais comum é o CI somador paralelo de quatro bits que

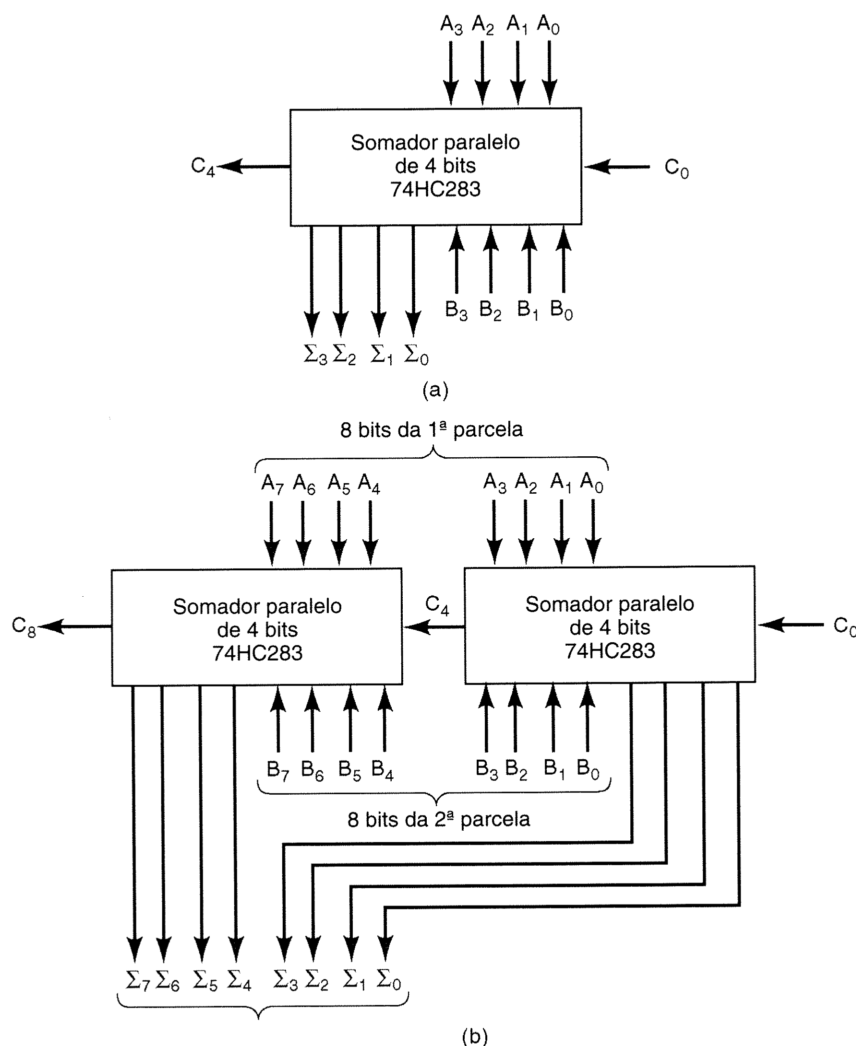


Fig. 6-10 - (a) Símbolo para o somador paralelo de quatro bits 74HC283. (b) Conectando dois 74HC283s.

contém quatro circuitos somadores completos e um circuito de carry antecipado que é necessário para operação em alta velocidade. Os 7483A, 74LS83A, 74283 e 74LS283 são todos chips TTL somadores paralelos de quatro bits. Os 283s são idênticos aos 83s, exceto que possuem V_{CC} e terra nos pinos 16 e 8, respectivamente. Isto se tornou um padrão em todos os novos chips que possuem os pinos de alimentação e terra nas extremidades do chip. O 74HC283 é a versão CMOS de alta velocidade para este somador paralelo de quatro bits.

A Fig. 6-10(a) mostra o símbolo funcional para o somador paralelo de quatro bits 74HC283 e seus equivalentes. As entradas deste CI são dois números de quatro bits $A_3A_2A_1A_0$ e $B_3B_2B_1B_0$ e o carry C_0 para a posição LSB. As saídas são os bits da soma e o carry C_4, proveniente da posição MSB. Os bits da soma são denominados $\Sigma_3\Sigma_2\Sigma_1\Sigma_0$, onde Σ é a letra grega *sigma* maiúscula. A denominação Σ é apenas uma alternativa comum à denominação S para o bit de soma.

Ligação em Cascata de Somadores Paralelos

Dois ou mais blocos somadores paralelos podem ser conectados para acomodar a adição de números binários maiores. Para ilustrar, a Fig. 6-10 (b) mostra como dois somadores 74HC283 podem ser conectados para somar dois números de oito bits. O somador à direita adiciona os quatro bits menos significativos dos números. A saída C_4 deste somador é conectada à entrada de carry da posição menos significativa do segundo somador, que adiciona os quatro bits mais significativos dos números. C_8 é o carry da última posição (MSB) do segundo somador. C_8 pode ser usado como um bit de overflow ou como um carry para um outro estágio somador se números binários maiores forem manipulados.

EXEMPLO 6-10

Determine os níveis lógicos nas entradas e saídas do somador de oito bits da Fig. 6-10 (b) quando 72_{10} é somado à 137_{10}.

Solução

Primeiro converta cada número para o equivalente em binário de oito bits:

$$137 = 10001001$$
$$72 = 01001000$$

Estes dois valores binários devem ser aplicados às entradas A e B, isto é, os bits da entrada A serão 10001001, da esquerda para a direita, e os bits da entrada B serão 01001000, da esquerda para a direita. O somador produzirá a soma binária dos dois números:

$$[A] = 10001001$$
$$[B] = \underline{01001000}$$
$$[\Sigma] = 11010001$$

Os bits da soma serão 11010001, da esquerda para a direita. Não existe overflow para o bit C_8, e, portanto, ele deverá ser igual a 0.

Questões de Revisão

1. Quantos chips 74HC283 são necessários para somar dois números de 20 bits?
2. Se um 74HC283 possui um atraso de propagação máximo de 30 ns de C_0 a C_4, qual será o atraso de propagação total de um somador de 32 bits construído de 74HC283s?

6-15 SISTEMA DE COMPLEMENTO A 2

A maioria dos computadores modernos usa o sistema de complemento a 2 para representar números negativos e para realizar a subtração. As operações de adição e subtração, de números com sinal, podem ser realizadas utilizando-se apenas a operação de adição, se usarmos o complemento a 2 para representar números negativos.

Adição

Números positivos e negativos, incluindo os bits de sinal, podem ser somados em um circuito somador paralelo básico, quando os números negativos estão representados em complemento a 2. Isto está ilustrado na Fig. 6-11 para a adição de -3 e $+6$. O -3 está representado em complemento a 2 como 1101, onde o primeiro 1 é o bit de sinal. O $+6$ está representado como 0110, com o primeiro 0 como bit de sinal. Estes números estão armazenados em seus registradores correspondentes. O somador paralelo de quatro bits produz como resultado 0011, que representa $+3$. A saída C_4 é igual a 1, mas lembre-se de que ela é desconsiderada no método do complemento a 2.

Subtração

Quando o complemento a 2 é usado, o número a ser subtraído (subtraendo) é substituído pelo seu complemento a 2, e depois é *somado* ao minuendo (o número do qual o subtraendo está sendo subtraído). Por exemplo, podemos considerar que o minuendo já está armazenado no acumulador (registrador A). O subtraendo é então colocado no registrador B (em um computador ele seria transferido da memória para o registrador) e é substituído pelo seu complemento a 2 antes de ser somado ao número que está no registrador A. Os bits de saída do circuito somador agora representam a *diferença* entre o minuendo e o subtraendo.

O circuito somador paralelo que estivemos estudando pode ser adaptado para realizar a subtração descrita anteriormente se fornecermos um modo de obter o complemento a 2 do número que está no registrador B. O complemento a 2 de um número binário é obtido através da complementação (inversão) de cada bit seguido da adição de 1 ao LSB. A Fig. 6-12 mostra como isto pode ser obtido. As saídas *invertidas* do registrador B são usadas no lugar das saídas normais, isto é, $\overline{B_0}$, $\overline{B_1}$, $\overline{B_2}$, e $\overline{B_3}$ são conectadas às entradas do somador (lembre-se de que B_3 é o bit de sinal). Isto resolve o problema de complementar cada bit do número B. Além disso, C_0 é colocado em 1, de modo que isto adiciona 1 extra na posição menos

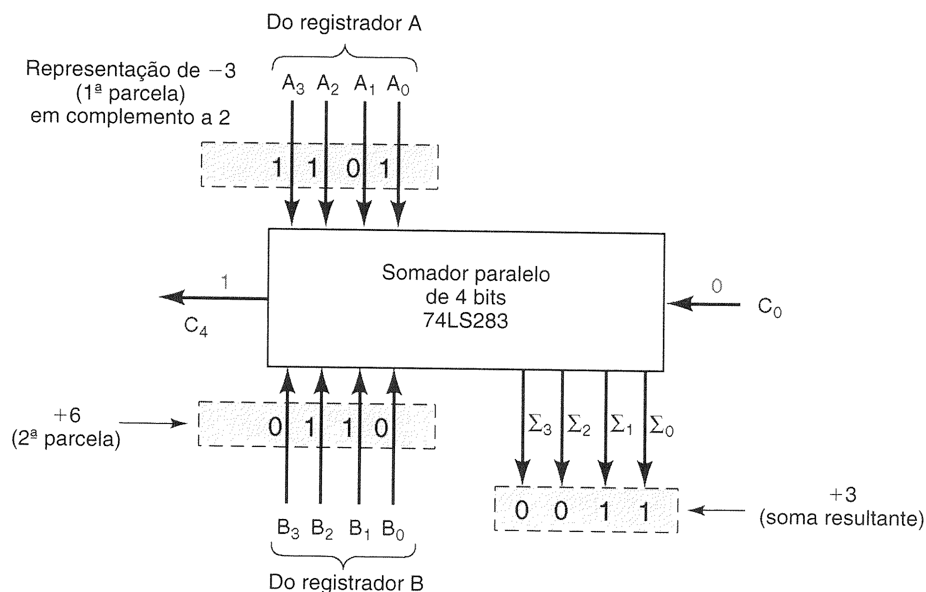


Fig.6-11 - Somador paralelo usado para adicionar e subtrair números no sistema de complemento a 2.

significativa do somador. Isto tem o mesmo efeito que adicionar 1 no LSB do registrador *B* para a obtenção do complemento a 2.

As saídas Σ_3 a Σ_0 representam o resultado da operação de subtração. É claro que Σ_3 é o bit de sinal do resultado e indica se este é positivo ou negativo. A saída carry C_4 é novamente desconsiderada.

Para ajudar você a entender esta operação, estude os passos para subtrair +6 de +4:

1. +4 é armazenado no registrador *A* como 0100.
2. +6 é armazenado no registrador *B* como 0110.
3. As saídas invertidas dos FFs do registrador *B* (1001) são conectadas ao somador.
4. O circuito somador paralelo soma $[A] = 0100$ a $[\overline{B}] = 1001$ junto com o carry $C_0 = 1$, no LSB. A operação está representada a seguir.

$$\begin{array}{rl} 1 & \leftarrow C_0 \\ 0100 & \leftarrow [A] \\ +\ \underline{1001} & \leftarrow [\overline{B}] \\ 1110 & \leftarrow [\Sigma] = [A] - [B] \end{array}$$

O resultado presente nos bits de soma é 1110. Ele, na verdade, representa o resultado da operação de *subtração*, a *diferença* entre o número no registrador *A* e aquele que está no registrador *B*, isto é, $[A] - [B]$. Uma vez que o bit de sinal é igual a 1, o resultado é negativo e está em complemento a 2. Podemos verificar que 1110 representa -2_{10} tomando seu complemento a 2 e obtendo $+2_{10}$:

$$\begin{array}{r} 1110 \\ 0001 \\ +\ \underline{\quad 1} \\ 0010 = +2_{10} \end{array}$$

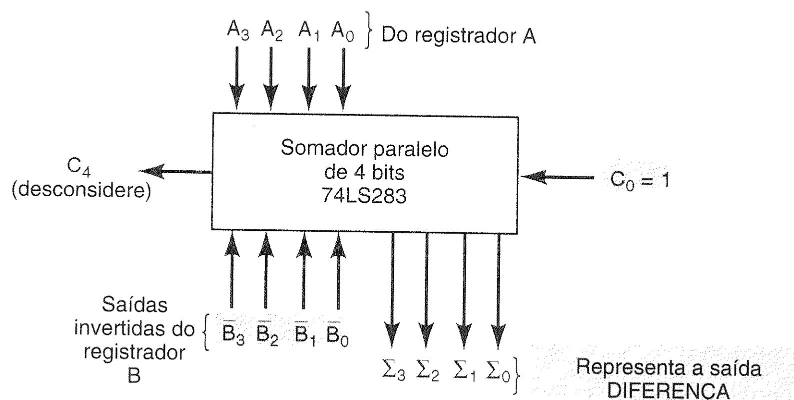


Fig. 6-12 - Somador paralelo utilizado para realizar a subtração $(A - B)$ usando o sistema de complemento a 2. Os bits do subtraendo (B) são invertidos e $C_0 = 1$ para produzir o complemento a 2.

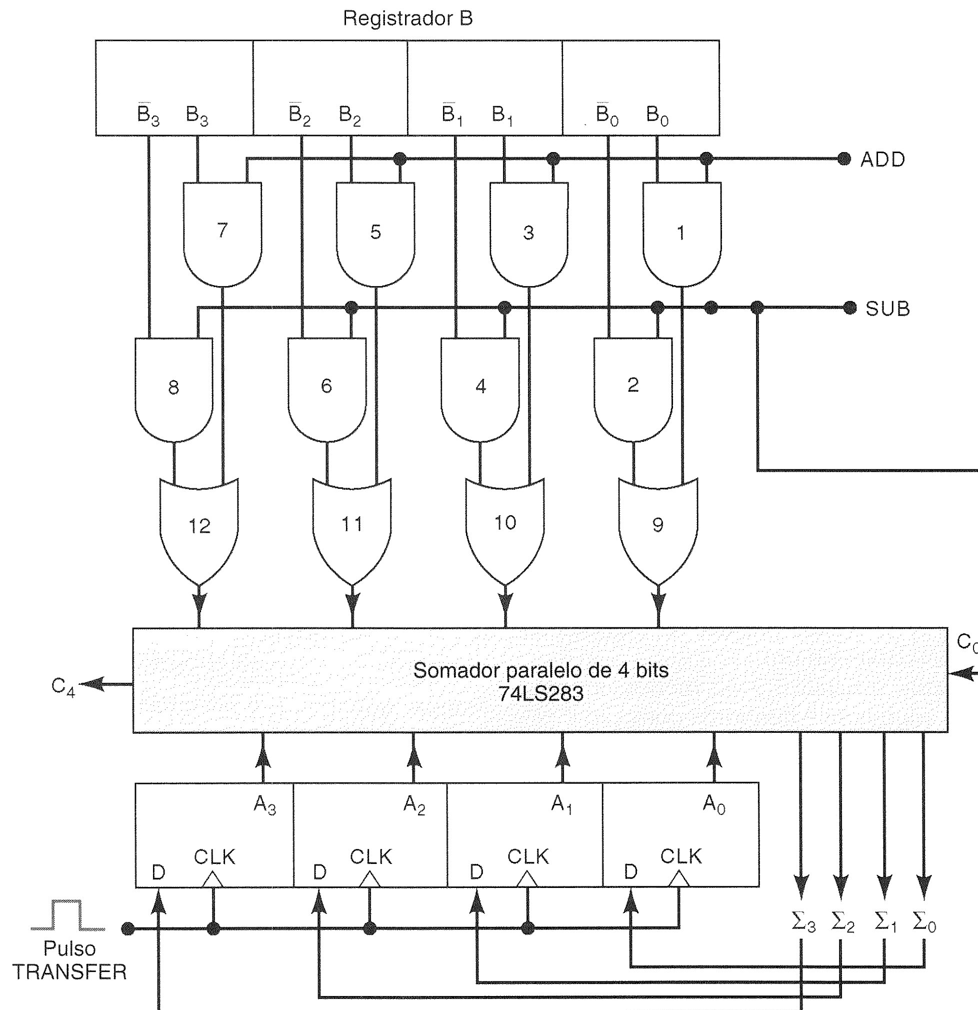


Fig. 6-13 - Somador/subtrator paralelo usando o sistema de complemento a 2.

Adição e Subtração Combinadas

Deve estar claro agora que o circuito somador paralelo básico pode ser usado para realizar adição ou subtração, dependendo de se o número *B* permanece inalterado ou se ele é convertido para o seu complemento a 2. Um circuito completo que realiza *tanto* a adição *como* a subtração em complemento a 2 é mostrado na Fig. 6-13.

Este circuito somador/subtrator é controlado por dois sinais, chamados ADD e SUB. Quando ADD está em ALTO, o circuito realiza a adição dos números armazenados nos registradores *A* e *B*. Quando SUB está em ALTO, o circuito subtrai o número que está em *B* daquele que está em *A*. A operação está descrita a seguir:

1. Suponha que ADD = 1 e SUB = 0. O sinal SUB = 0 *desabilita* (inibe) as portas AND 2, 4, 6 e 8, mantendo suas saídas em 0. O sinal ADD = 1 *habilita* as portas 1, 3, 5 e 7, permitindo que suas saídas tenham os níveis de B_0, B_1, B_2 e B_3, respectivamente.
2. Os níveis B_0 a B_3 passam através das portas OR para o somador paralelo de quatro bits para serem somados com os bits A_0 a A_3. Esta *soma* aparece nas saídas Σ_3 a Σ_0.
3. Observe que SUB = 0 faz com que o carry C_0 seja igual a 0.
4. Agora suponha que ADD = 0 e SUB = 1. O sinal ADD = 0 inibe as portas AND 1, 3, 5 e 7. O sinal SUB habilita as portas AND 2, 4, 6 e 8, de modo que em suas saídas aparecem os níveis de $\overline{B}_0$, $\overline{B}_1$, $\overline{B}_2$, e $\overline{B}_3$ respectivamente.
5. Os níveis de $\overline{B}_0$ a $\overline{B}_3$ passam pelas portas OR e vão para o somador para serem somados aos bits A_0 a A_3. Note também que, agora, C_0 é igual a 1. Portanto, o número que está no registrador *B* foi convertido para o seu complemento a 2.
6. A *diferença* aparece nas *saídas* Σ_3 a Σ_0.

Circuitos como o somador/subtrator da Fig. 6-13 são usados em computadores porque fornecem um meio relativamente simples de somar e subtrair números binários com sinal. Na maioria dos computadores, os valores presentes nas linhas de saída Σ são, geralmente, transferidos para o registrador *A* (acumulador), de modo que os resultados da adição ou da subtração terminam sempre armazenados no registrador *A*. Isto é obtido através da aplicação de um pulso TRANSFER nas entradas CLK do registrador *A*.

Questões de Revisão

1. Por que C_0 deve ser igual a 1 para que o circuito somador da Fig. 6-12 possa ser usado como um subtrator?
2. Suponha que $[A]$ = 0011 e $[B]$ = 0010 na Fig. 6-13. Se ADD = 1 e SUB = 0, determine os níveis lógicos nas saídas das portas OR.
3. Repita a questão 2 para ADD = 0 e SUB = 1.
4. *Verdadeiro ou falso*: Quando o circuito somador/subtrator é usado para subtração, o complemento a 2 do subtraendo aparece na entrada do somador.

TABELA 6-3

S_4	S_3	S_2	S_1	S_0	
0	1	0	1	0	(10)
0	1	0	1	1	(11)
0	1	1	0	0	(12)
0	1	1	0	1	(13)
0	1	1	1	0	(14)
0	1	1	1	1	(15)
1	0	0	0	0	(16)
1	0	0	0	1	(17)
1	0	0	1	0	(18)

6-16 SOMADOR BCD

O processo de adição BCD foi discutido na Seção 6-7 e é revisto a seguir:

1. Some os códigos BCD para cada posição de dígito decimal; utilize a soma binária comum.
2. Para as posições onde a soma é 9 ou menos, a soma está em forma BCD adequada e nenhuma correção é necessária.
3. Quando a soma de dois dígitos é maior do que 9, uma correção de 0110 deve ser adicionada, de modo a produzir um resultado BCD apropriado. Isto produzirá um carry a ser somado na posição decimal seguinte.

Um circuito somador BCD deve estar apto a operar de acordo com os passos anteriormente descritos. Em outras palavras, o circuito deve ser capaz de fazer o seguinte:

1. Somar dois códigos BCD de quatro bits utilizando a soma binária direta.
2. Determinar se o resultado desta adição é maior do que 1001 (9 decimal); se for, somar 0110 (6) ao resultado e gerar um carry para a próxima posição decimal.

O primeiro requisito é facilmente atendido utilizando-se um somador binário paralelo de quatro bits, tal como o 74HC283 e seus equivalentes. Por exemplo, se dois códigos BCD representados por $A_3A_2A_1A_0$ e $B_3B_2B_1B_0$, respectivamente, são aplicados em um somador binário paralelo de quatro bits, o somador realizará a seguinte operação:

$$\begin{array}{rl} A_3A_2A_1A_0 & \leftarrow \text{código BCD} \\ +\ \underline{B_3B_2B_1B_0} & \leftarrow \text{código BCD} \\ S_4S_3S_2S_1S_0 & \leftarrow \text{soma binária comum} \end{array}$$

S_4 é na verdade C_4, o carry vindo do MSB.

As saídas da soma $S_4S_3S_2S_1S_0$ podem variar desde 00000 até 10010 (quando os dois códigos BCD são 1001 = 9). O circuito para um somador BCD deve incluir a lógica necessária para detectar sempre que a soma for maior do que 01001, de modo que a correção possa ser adicionada. Estes casos, onde a soma é maior do que 01001, estão relacionados na Tabela 6-3. Vamos definir X como uma saída lógica que vai para ALTO somente quando a soma é maior do que 01001 (isto é, para os casos relacionados na Tabela 6-3). Se examinarmos estes casos, pode ser constatado que X vai para ALTO para qualquer das condições a seguir:

1. Sempre que S_4 = 1 (somas maiores do que 15)
2. Sempre que $S3$ = 1 e ou S_2 ou S_1 ou ambos estão em 1 (somas entre 10 e 15)

Isto pode ser expresso como

$$X = S_4 + S_3(S_2 + S_1)$$

Sempre que $X = 1$, é necessário adicionar a correção 0110 à soma de bits e gerar um carry. A Fig. 6-14 mostra o circuito completo para o somador BCD, incluindo o circuito lógico para a implementação de X.

O circuito é composto de três partes básicas. Os dois códigos $A_3A_2A_1A_0$ e $B_3B_2B_1B_0$ são adicionados ao somador de quatro bits superior para produzir a soma $S_4S_3S_2S_1S_0$. As portas lógicas implementam a expressão para X. O somador de quatro bits inferior adiciona a correção 0110 na soma de bits *somente* quando $X = 1$, produzindo a saída final da soma BCD representada por $\Sigma_3\Sigma_2\Sigma_1\Sigma_0$. X também é a saída de carry que é produzida quando a soma é maior do que 01001. Naturalmente, quando $X = 0$, não existe carry e nenhuma adição de 0110. Nesses casos, $\Sigma_3\Sigma_2\Sigma_1\Sigma_0 = S_3S_2S_1S_0$.

Para auxiliar no entendimento do somador BCD, o estudante deveria experimentar vários casos seguindo-os no circuito. Os casos seguintes poderiam ser especialmente instrutivos:

Entradas

(a) $[A]$ = 0101, $[B]$ = 0011, C_0 = 0
(b) $[A]$ = 0111, $[B]$ = 0110, C_0 = 0

Saídas

(a) $[S]$ = 01000, X = 0, $[\Sigma]$ = 1000, CARRY = 0
(b) $[S]$ = 01101, X = 1, $[\Sigma]$ = 0011, CARRY = 1

Ligação em Cascata de Somadores BCD

O circuito da Fig. 6-14 é utilizado para somar dois dígitos decimais que foram codificados em BCD. Quando números decimais de vários dígitos devem ser somados, é necessário usar um somador BCD separado para cada posição de dígito. A Fig. 6-15 é um diagrama de blocos de um circuito para a adição de dois números decimais de três dígitos. O registrador A contém 12 bits, que são os três códigos BCD para um dos números decimais de três dígitos; analogamente, o registrador B contém a representação BCD do

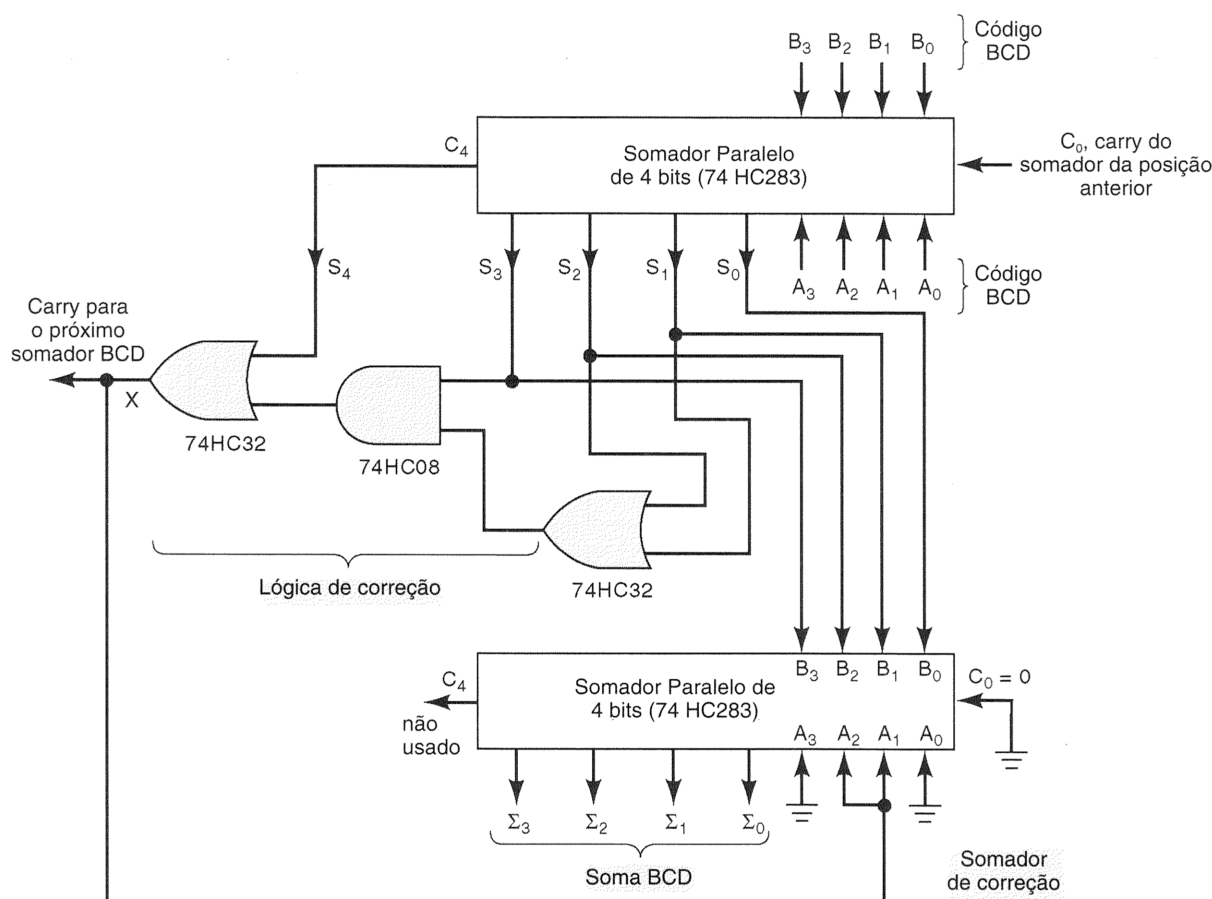


Fig. 6-14 - Um somador BCD contém dois somadores de quatro bits e um circuito detector para correção.

outro número decimal de três dígitos. Os grupos de código A_3-A_0 e B_3-B_0 representando os dígitos menos significativos são conectados ao primeiro somador BCD. Cada bloco somador BCD contém o circuito da Fig. 6-14. Esse primeiro somador BCD produz as saídas de soma $\Sigma_3\Sigma_2\Sigma_1\Sigma_0$, que é o código BCD para o dígito menos significativo da soma. Ele também produz uma saída de carry que é enviada para o segundo somador BCD, que está somando de A_7 a A_4 com B_7 a B_4, os grupos de código para a segunda posição de dígito decimal. O segundo somador BCD produz $\Sigma_7\Sigma_6\Sigma_5\Sigma_4$, o código BCD para o segundo dígito da soma, e assim por diante. Esse arranjo pode, naturalmente, ser estendido para números decimais de qualquer tamanho simplesmente acrescentando-se mais FFs aos registradores e incluindo-se um somador BCD para cada posição de dígito.

EXEMPLO 6-11

Determine as entradas e saídas quando o circuito da Fig. 6-15 é usado para somar 538_{10} com 247_{10}.

Solução

Primeiramente, os números decimais são representados em BCD.

247 = 0010 0100 0111 (BCD)
538 = 0101 0011 1000 (BCD)

Esses números BCD serão colocados nos registradores *A* e *B*, respectivamente, de modo que

[*A*] = 0010 0100 0111
[*B*] = 0101 0011 1000

O CARRY IN para o somador LSD será 0.

Uma vez que os dados estão nos registradores, os somadores BCD começarão a produzir as somas BCD corretas em suas saídas. O somador LSD adicionará o 0111 (7) com 1000 (8) para produzir uma soma de 0101 (5) e um CARRY de 1 para o somador intermediário. O somador intermediário adicionará o 0100 (4) com o 0011 (3) e o CARRY de 1 para produzir uma soma de 1000 (8) e um CARRY de 0 para o somador MSD. O somador MSD adicionará 0010 (2) com 0101 (5) para uma soma de 0111 (7) e nenhum CARRY na saída. Assim, nas saídas de soma temos

$[\Sigma]$ = 0111 1000 0101

e existe uma saída de CARRY de 0 do somador MSD. Esse resultado é, obviamente, a representação BCD da soma decimal 785_{10}.

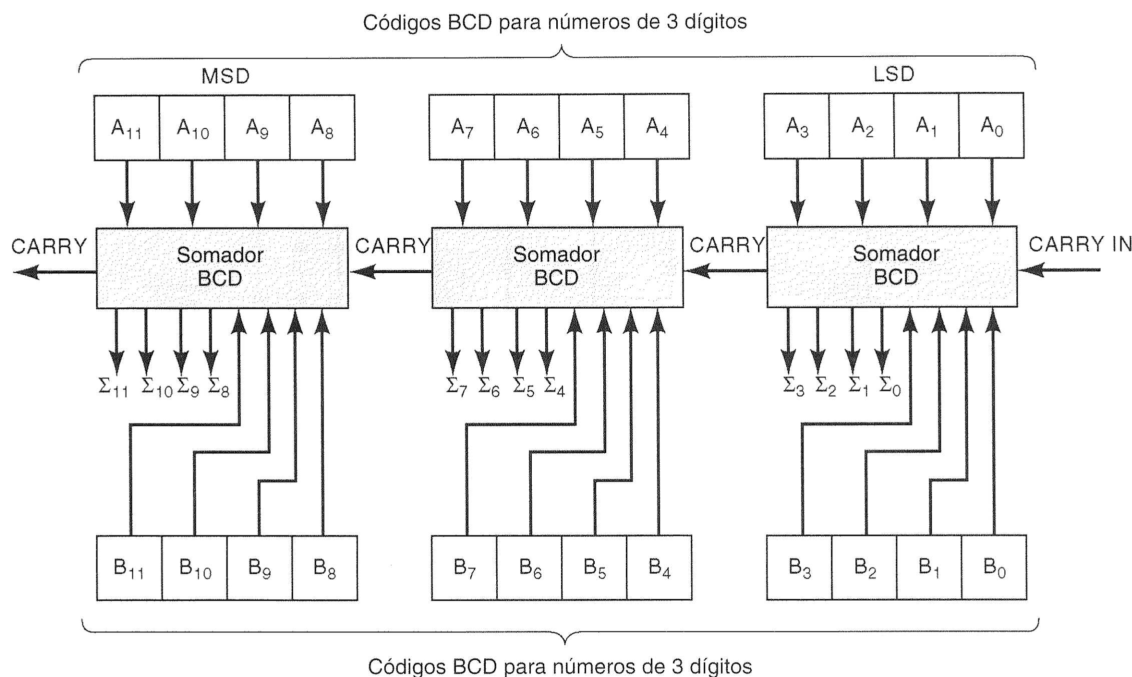


Fig. 6-15 - Ligação em cascata de somadores BCD para adicionar dois números decimais de três dígitos.

Questões de Revisão

1. Quais são as três partes básicas de um circuito somador BCD?
2. Descreva como o circuito somador BCD detecta a necessidade de uma correção e a executa.

6-17 CIRCUITOS INTEGRADOS DE ULAs

Existem vários circuitos integrados disponíveis que são denominados de unidades lógicas e aritméticas (ULAs) mesmo não tendo as capacidades completas das unidades lógicas e aritméticas dos computadores. Estes chips de ULA são capazes de realizar muitas operações lógicas e aritméticas diferentes sobre os dados binários de entrada. A operação específica que um CI de ULA realiza é determinada por um código binário aplicado nas entradas selecionadoras de função. Alguns CIs de ULA são bastante complexos, e demandariam muito tempo e espaço para explicar e ilustrar suas operações. Nesta seção usaremos um chip de ULA relativamente simples, embora útil, para mostrar os conceitos básicos envolvidos nos CIs de ULA. As idéias apresentadas podem ser estendidas para dispositivos mais complexos.

A ULA 74LS382/HC382

A Fig. 6-16(a) mostra o símbolo para uma ULA que está disponível como um 74LS382 (TTL) e como um 74HC382 (CMOS). Este CI de 20 pinos opera sobre dois números de entrada de quatro bits, $A_3A_2A_1A_0$ e $B_3B_2B_1B_0$, para produzir um resultado de quatro bits $F_3F_2F_1F_0$. Essa ULA pode realizar *oito* operações diferentes. Em determinado instante, a operação que ela está realizando depende do código de entrada aplicado nas entradas selecionadoras de função $S_2S_1S_0$. A tabela na Fig. 6-16 (b) mostra as oito operações disponíveis. Descreveremos agora cada uma destas operações.

OPERAÇÃO CLEAR Com $S_2S_1S_0$ = 000, a ULA vai *limpar* todos os bits das saídas *F* de modo que $F_3F_2F_1F_0$ = 0000.

OPERAÇÃO SOMA Com $S_2S_1S_0$ = 011, a ULA adicionará $A_3A_2A_1A_0$ e $B_3B_2B_1B_0$ para produzir a soma em $F_3F_2F_1F_0$. Para esta operação, C_N é o carry de entrada na posição LSB, e ele deve ser mantido em 0. C_{N+4} é a saída de carry da posição MSB. *OVR* é a saída indicadora de overflow; ela detecta overflow quando números com sinal estão sendo usados. *OVR* será 1 quando uma operação de soma ou de subtração produzir um resultado que é muito grande para ser representado com quatro bits (incluindo o bit de sinal).

OPERAÇÕES DE SUBTRAÇÃO Com $S_2S_1S_0$ = 001, a ULA subtrairá o número da entrada *A* do número da entrada *B*. Com $S_2S_1S_0$ = 010, a ULA subtrairá *B* de *A*. Em ambos os casos, a diferença aparece em $F_3F_2F_1F_0$. Note que as operações de subtração necessitam que a entrada C_N esteja em 1.

OPERAÇÃO EX-OR Com $S_2S_1S_0$ = 100, a ULA realizará uma operação EX-OR bit a bit nas entradas *A* e *B*. Isto é ilustrado a seguir para $A_3A_2A_1A_0$ = 0110 e $B_3B_2B_1B_0$ = 1100.

$$A_3 \oplus B_3 = 0 \oplus 1 = 1 = F_3$$
$$A_2 \oplus B_2 = 1 \oplus 1 = 0 = F_2$$
$$A_1 \oplus B_1 = 1 \oplus 0 = 1 = F_1$$
$$A_0 \oplus B_0 = 0 \oplus 0 = 0 = F_0$$

O resultado é $F_3F_2F_1F_0$ = 1010.

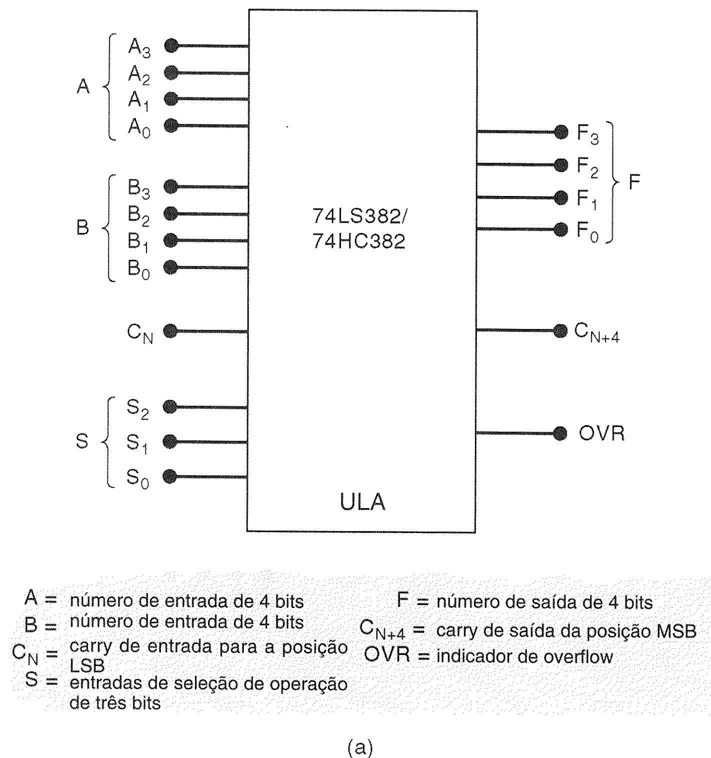


Tabela Funcional

S_2	S_1	S_0	Operação	Comentários
0	0	0	CLEAR	$F_3F_2F_1F_0 = 0000$
0	0	1	B menos A	} Necessita $C_N = 1$
0	1	0	A menos B	
0	1	1	A mais B	Necessita $C_N = 0$
1	0	0	$A \oplus B$	Exclusive-OR
1	0	1	A + B	OR
1	1	0	AB	AND
1	1	1	PRESET	$F_3F_2F_1F_0 = 1111$

Notas: As entradas S selecionam a operação.
OVR = 1 para overflow com números com sinal.

(b)

Fig. 6-16 - (a) Símbolo para o chip ULA 74LS382/74HC382; (b) tabela funcional que mostra como as entradas de seleção (S) determinam qual operação deve ser realizada sobre as entradas A e B.

OPERAÇÃO OR Com $S_2S_1S_0 = 101$, a ULA realizará uma operação OR bit a bit nas entradas A e B. Por exemplo, com $A_3A_2A_1A_0 = 0110$ e $B_3B_2B_1B_0 = 1100$, a ULA gerará um resultado $F_3F_2F_1F_0 = 1110$.

OPERAÇÃO AND Com $S_2S_1S_0 = 110$, a ULA realizará uma operação AND bit a bit nas entradas A e B. Por exemplo, com $A_3A_2A_1A_0 = 0110$ e $B_3B_2B_1B_0 = 1100$, a ULA gerará um resultado $F_3F_2F_1F_0 = 0100$.

OPERAÇÕES PRESET Com $S_2S_1S_0 = 111$, a ULA *colocará* todos os bits da saída em 1, logo $F_3F_2F_1F_0 = 1111$.

EXEMPLO 6-12

(a) Determine as saídas do 74HC382 para as seguintes entradas: $S_2S_1S_0 = 010$, $A_3A_2A_1A_0 = 0100$, $B_3B_2B_1B_0 = 0001$ e $C_N = 1$.
(b) Substitua o código de seleção por 011 e repita o item (a).

Solução

(a) Da tabela funcional na Fig. 6-16 (b), 010 seleciona a operação ($A - B$). A ULA realizará a subtração por complemento a 2 complementando B e somando-o com A e C_N. Note que $C_N = 1$ é necessário para efetivamente completar o complemento a 2 de B.

$$\begin{array}{rl} 1 & \leftarrow C_N \\ 0100 & \leftarrow A \\ +\ \underline{1110} & \leftarrow \overline{B} \\ 10011 & \\ C_{N+4} \uparrow \quad \uparrow & F_3F_2F_1F_0 \end{array}$$

Como sempre na subtração de complemento a 2, o CARRY de saída é descartado. O resultado correto da operação ($A - B$) aparece nas saídas F.

A saída OVR é determinada considerando que os números de entrada são números com sinal. Assim, temos $A_3A_2A_1A_0 = 0100 = +4_{10}$ e $B_3B_2B_1B_0 = 0001 = +1_{10}$. O resultado da operação de subtração é $F_3F_2F_1F_0 = 0011 = +3_{10}$, que está correto. Portanto, não ocorreu overflow, e $OVR = 0$. Se o resultado tivesse sido negativo, estaria na forma de complemento a 2.

(b) Um código de seleção de 011 produzirá a soma das entradas A e B. Entretanto, visto que $C_N = 1$, existirá um carry de 1 adicionado na posição do LSB. Isto produzirá um resultado de $F_3F_2F_1F_0 = 0110$, que é maior uma unidade do que ($A + B$). As saídas C_{N+4} e OVR estarão ambas em 0. Para a soma correta aparecer em F, a entrada C_N deve ser 0.

Expandindo a ULA

Um 74LS382 ou 74HC382 opera em números de quatro bits. Dois ou mais chips podem ser conectados para operar sobre números maiores. A Fig. 6-17 mostra como duas ULAs

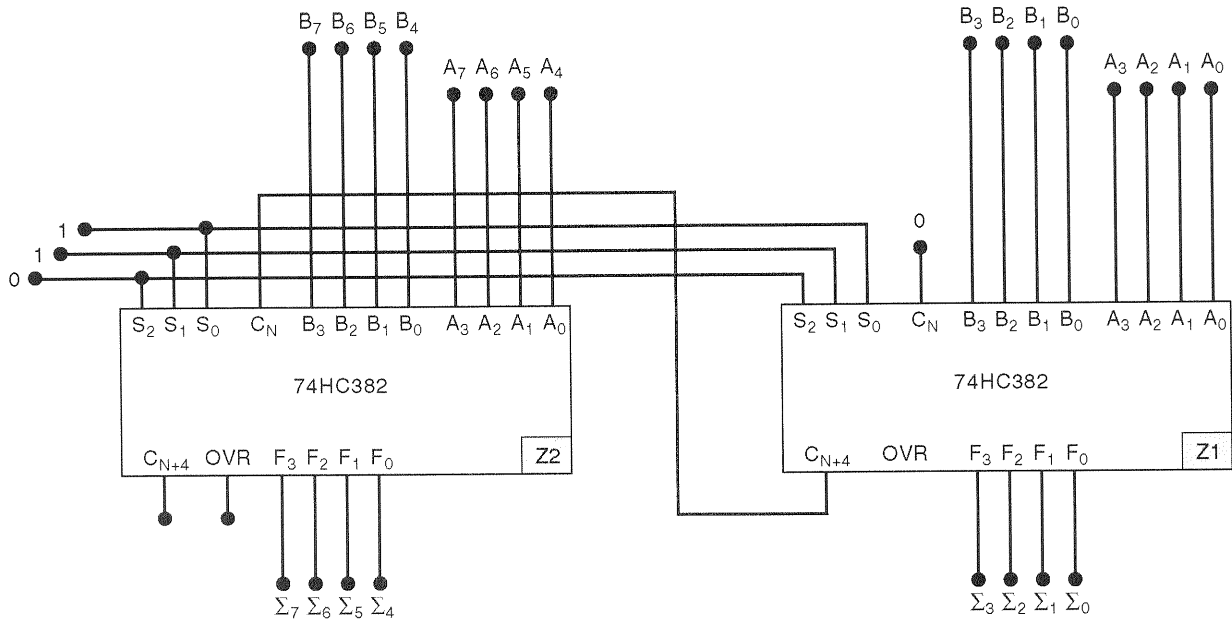


Fig. 6-17 - Dois chips de ULA 74HC382 conectados como um somador de oito bits.

de quatro bits podem ser combinadas para somar dois números de oito bits, $B_7B_6B_5B_4B_3B_2B_1B_0$ e $A_7A_6A_5A_4A_3A_2A_1A_0$, para produzir a soma de saída $\Sigma_7\Sigma_6\Sigma_5\Sigma_4\Sigma_3\Sigma_2\Sigma_1\Sigma_0$. Estude o diagrama do circuito e observe os seguintes pontos:

1. O chip Z1 opera nos quatro bits de mais baixa ordem dos dois números de entrada. O chip Z2 opera nos quatro bits de mais alta ordem.
2. A soma aparece nas saídas *F* de Z1 e Z2. Os bits de mais baixa ordem aparecem em Z1, e os de mais alta ordem, em Z2.
3. A entrada C_N de Z1 é o carry para a posição LSB. Para a adição, ele é colocado em 0.
4. A saída de carry [C_{N+4}] de Z1 é conectada na entrada de carry [C_N] de Z2.
5. A saída *OVR* de Z2 é o indicador de overflow quando números com sinal de oito bits estão sendo usados.
6. As entradas de seleção correspondentes dos dois chips são conectadas juntas, de modo que Z1 e Z2 estão sempre realizando a mesma operação. Para a adição, as entradas de seleção estão com 011.

EXEMPLO 6-13

Como o arranjo da Fig. 6-17 teria que ser alterado de modo a realizar a subtração ($B - A$)?

Solução

O código da entrada de seleção [veja a tabela na Fig. 6-16(b)] deve ser alterado para 001, e a entrada C_N de Z1 deve ser 1.

Outras ULAs

O 74LS181/HC181 é uma outra ULA de quatro bits. Ele tem quatro entradas que selecionam qualquer uma das 16 operações diferentes. Ele também possui um bit de entrada que permite selecionar entre operações lógicas e operações aritméticas (soma e subtração). Esta ULA tem uma saída $A = B$ que é utilizada para comparar as magnitudes das entradas *A* e *B*. Quando os dois números de entrada são exatamente iguais, a saída $A = B$ fica em 1; caso contrário, ela é 0.

O 74LS881/HC881 é similar ao chip 181, mas tem a capacidade de realizar algumas operações lógicas adicionais.

Questões de Revisão

1. Aplique as seguintes entradas na ULA da Fig. 6-16 e determine as saídas: $S_2S_1S_0 = 001$, $A_3A_2A_1A_0 = 1110$, $B_3B_2B_1B_0 = 1001$, $C_N = 1$.
2. Troque o código de seleção para 011 e C_N para 0 e repita a questão 1.
3. Troque o código de seleção para 110 e repita a questão 1.
4. Aplique as seguintes entradas no circuito da Fig. 6-17 e determine as saídas: $B = 01010011$, $A = 00011000$.
5. Troque o código de seleção para 111 e repita a questão 4.
6. Quantos 74HC382 são necessários para somar dois números de 32 bits?

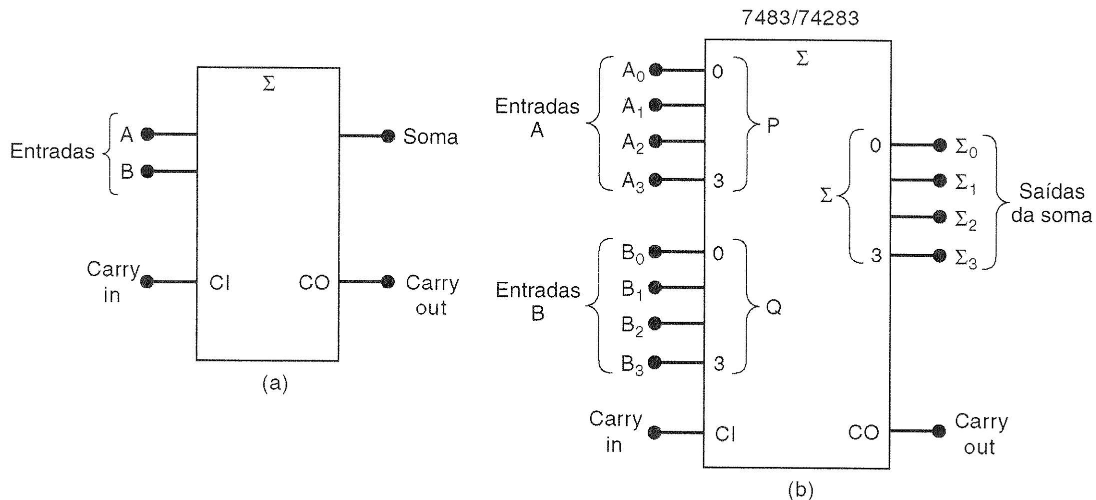


Fig. 6-18 - Símbolos IEEE/ANSI para (a) um somador completo e (b) um CI somador paralelo de quatro bits (7483/74283).

6-18 SÍMBOLOS IEEE/ANSI

A Fig. 6-18 (a) mostra o símbolo IEEE/ANSI para um somador de um bit (somador completo). Note que o símbolo Σ é usado para indicar a operação de adição. A Fig. 6-18 (b) é o símbolo IEEE/ANSI para o somador paralelo de quatro bits 7483/74283. Note que as letras *P* e *Q* são usadas para representar as duas entradas de quatro bits e Σ é utilizado para a soma de saída de quatro bits. *P*, *Q* e Σ estão especificados pelo padrão IEEE/ANSI e devem ser usados dentro do símbolo. As letras usadas para as entradas e saídas externas ao símbolo não estão especificadas pelo padrão.

6-19 ESTUDO DE CASO EM PESQUISA DE FALHAS

Uma estudante está testando o somador/subtrator apresentado na Fig. 6-19 e registra os seguintes resultados de teste para os vários modos de operação:

- **Modo 1: ADD = 0, SUB = 0.** As saídas de soma são sempre iguais ao número no registrador *A mais um*. Por exemplo, quando [*A*] = 0110, a soma é [Σ] = 0111. Isto está incorreto, visto que as saídas das portas OR e C_0 deveriam ser todas 0 neste modo para produzir [Σ] = [*A*].
- **Modo 2: ADD = 1, SUB = 0.** A soma é sempre um a mais do que deveria ser. Por exemplo, com [*A*] = 0010 e [*B*] = 0100, a saída é 0111 em vez de 0110.
- **Modo 3: ADD = 0, SUB = 1.** As saídas Σ são sempre iguais a [*A*] − [*B*], como esperado.

Quando examina esses resultados, a estudante constata que as saídas de soma superam os valores esperados de uma unidade para os dois primeiros modos de operação. Primeiro ela suspeita de uma possível falha em uma das entradas LSB do somador, mas ela descarta isso, pois tal falha afetaria também a operação de subtração, que está funcionando corretamente. Casualmente, ela percebe que existe uma outra falha que poderia somar 1 ao resultado para os dois primeiros modos sem causar um erro no modo de subtração.

Lembre-se de que C_0 deve ser 1 no modo subtração como parte da operação de complemento a 2 em [*B*]. Para os outros modos, C_0 deve ser 0. A estudante verifica a conexão entre o sinal *SUB* e a entrada C_0 do somador e constata que ela está aberta devido a uma solda malfeita. Esta conexão aberta explica os resultados observados, já que o somador TTL responde como se C_0 estivesse com nível lógico 1, causando que 1 fosse somado ao resultado nos modos 1 e 2. A conexão aberta não teria efeito no modo 3, pois C_0 deveria ser 1 mesmo.

EXEMPLO 6-14

Considere novamente o circuito somador/subtrator. Suponha que existe uma interrupção na ligação entre a entrada SUB e as portas AND no ponto *X* da Fig. 6-19. Descreva os efeitos disto na operação do circuito para cada modo.

Solução

Primeiramente, constate que esta falha produzirá um nível lógico 1, nas entradas 2, 4, 6 e 8 das portas AND, que habilitará permanentemente cada uma das portas a passar sua entrada $\overline{B}$ para a porta OR seguinte como ilustrado.

- **Modo 1: ADD = 0, SUB = 0.** A falha fará com que o circuito realize sempre uma subtração. O complemento a 1 de [*B*] alcançará as saídas das portas OR e será aplicado ao somador assim como [*A*]. Com C_0 = 0, o complemento a 2 de [*B*] não estará completo; não haverá a soma com 1. Deste modo, o somador produzirá [*A*] − [*B*] − 1. Para ilustrar, vamos tentar com [*A*] = +6 = 0110 e [*B*] = +3 = 0011. O somador adicionará da seguinte maneira:

Complemento a 1 de [*B*] =	1100
[*A*] =	0110
resultado =	10010

↑ Descartar o carry.

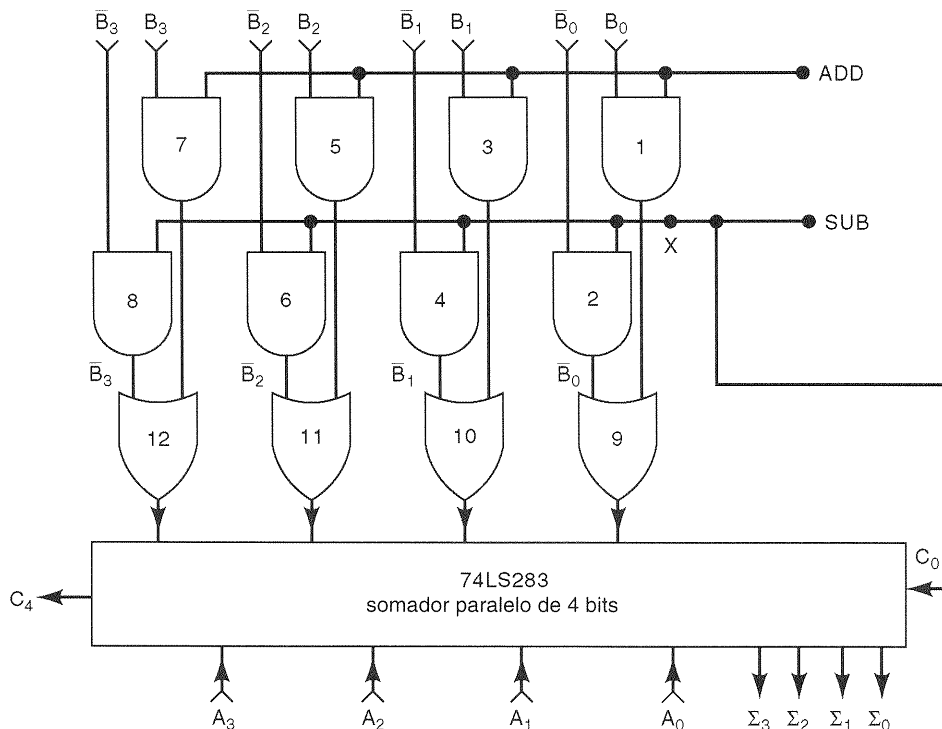


Fig. 6-19 - Circuito somador/subtrator paralelo.

O resultado é 0010 = +2 em vez de 0011 = +3, como seria normal para a subtração.

■ **Modo 2: ADD = 1, SUB = 0.** Com ADD = 1, as portas AND 1, 3, 5 e 7 passarão as entradas *B* para a porta OR seguinte. Assim, cada porta OR terá um $\overline{B}$ e um *B* nas suas entradas, produzindo uma saída 1. Por exemplo, as entradas, para a porta OR 9 serão $\overline{B}_0$, vindo da porta AND 2 (por causa da falha), e B_0, vindo da porta AND 1 (pois ADD = 1). Logo, a porta OR 9 produzirá uma saída $\overline{B}_0 + B_0$ que sempre será um nível lógico 1.

O somador adicionará 1111 das portas OR com [*A*] para produzir uma soma 1 unidade inferior a [*A*]. Por quê? Porque $1111_2 = -1_{10}$.

■ **Modo 3: ADD = 0, SUB = 1.** Este modo vai operar corretamente, tendo em vista que SUB = 1 habilitaria as portas AND 2, 4, 6 e 8 do mesmo jeito.

RESUMO

1. Para representar números com sinal em binário, um bit de sinal é anexado como o MSB. Para o sinal positivo o bit é 0, e para o negativo, é 1.
2. O complemento a 2 de um número binário é obtido complementando-se cada bit e então adicionando-se 1 ao resultado.
3. No método de complemento a 2 de representar números binários com sinal, números positivos são representados por um bit de sinal 0 seguido pela magnitude em sua forma binária direta. Números negativos são representados por um bit de sinal 1 seguido pela magnitude na forma de complemento a 2.
4. Um número binário com sinal é negado (alterado para um número de valor igual mas com sinal contrário) tomando-se o complemento a 2 do número, incluindo o bit de sinal.
5. A subtração pode ser realizada com números binários com sinal negando-se (gerando-se o complemento a 2) o subtraendo e somando-o ao minuendo.
6. Na adição BCD, um passo especial de correção é necessário sempre que a soma de uma certa posição exceder 9 (1001).
7. Quando números binários com sinal são representados em hexadecimal, o MSD do número hexa será 8 ou maior quando o número é negativo; ele será 7 ou menor quando o número é positivo.
8. A unidade lógica e aritmética (ULA) de um computador contém os circuitos necessários para a realização das operações lógicas e aritméticas com números binários armazenados na memória.
9. O acumulador é um registrador na ULA. Ele contém um dos operandos, e também é o local onde o resultado da operação é armazenado na ULA.
10. Um somador completo realiza a adição de dois bits e de uma entrada de carry. Um somador binário paralelo é feito ligando-se somadores completos em cascata.
11. O problema de atraso excessivo causado pela propagação do carry pode ser reduzido por um circuito lógico do tipo carry antecipado.
12. CIs somadores tais como 74LS83/HC83 e o 74LS283/HC283 podem ser utilizados para construir somadores e subtratores paralelos de alta velocidade.
13. Um circuito somador BCD requer circuito especial de correção.
14. Circuitos integrados de ULAs estão disponíveis e podem ser comandados a realizar uma ampla faixa de operações lógicas e aritméticas com dois números de entrada.

TERMOS IMPORTANTES

carry
bit de sinal
sistema sinal-magnitude
sistema de complemento a 2
negação
primeira parcela
segunda parcela
subtraendo
minuendo
overflow
unidade lógica e aritmética (ULA)
registrador acumulador
somador completo (SC)
somador/subtrator paralelo
meio somador (MS)
propagação do carry
carry antecipado

PROBLEMAS

SEÇÃO 6-1

6-1. Faça as somas seguintes em binário. Verifique os resultados fazendo a adição em decimal.

(a) 1010 + 1011
(b) 1111 + 0011
(c) 1011,1101 + 11,1
(d) 0,1011 + 0,1111
(e) 10011011 + 10011101

SEÇÃO 6-2

6-2. Represente cada um dos seguintes números decimais com sinal no sistema de complemento a 2. Utilize um total de oito bits incluindo o bit de sinal.

(a) +32
(b) −14
(c) +63
(d) −104
(e) +127
(f) −127
(g) +89
(h) −55
(i) −1
(j) −128
(k) +169
(l) 0

6-3. Cada um dos números seguintes representa um número decimal com sinal no sistema de complemento a 2. Determine o valor decimal de cada um. (*Sugestão:* Utilize a negação para converter números negativos em positivos.)

(a) 01101
(b) 11101
(c) 01111011
(d) 10011001
(e) 01111111
(f) 10000000
(g) 11111111
(h) 10000001
(i) 01100011
(j) 11011001

6-4. **(a)** Qual faixa de valores decimais com sinal pode ser representada usando 12 bits incluindo o bit de sinal?
(b) Quantos bits seriam necessários para representar números decimais de −32.768 até +32.767?

6-5. Relacione, em ordem, todos os números com sinal que podem ser representados com cinco bits usando o sistema de complemento a 2.

6-6. Represente cada um dos valores decimais a seguir como um valor binário de cinco bits com sinal. Depois negue cada um.

(a) +7 (b) −12 (c) +15 (d) −1

6-7. Qual é a faixa de valores decimais sem sinal que pode ser representada com 10 bits? Qual é a faixa de valores decimais com sinal usando o mesmo número de bits?

SEÇÕES 6-3 E 6-4

6-8. A razão pela qual o método de sinal-magnitude para representação de números com sinal não é usado na maioria dos computadores pode ser prontamente ilustrada realizando o seguinte.
(a) Represente +12 com cinco bits usando a forma sinal-magnitude.
(b) Represente −12 com cinco bits usando a forma sinal-magnitude.
(c) Some os dois números binários e note que a soma não se parece nada com zero.

6-9. Realize as seguintes operações no sistema de complemento a 2. Use oito bits (incluindo o bit de sinal) para cada número. Verifique os resultados convertendo o resultado binário de volta para decimal.

(a) Some + 9 com + 6.
(b) Some + 14 com − 17.
(c) Some + 19 com − 24.
(d) Some − 48 com − 80.
(e) Subtraia + 16 de + 17.
(f) Subtraia + 21 de − 13.
(g) Subtraia + 47 de + 47.
(h) Subtraia − 36 de − 15.
(i) Some + 17 com − 17.
(j) Subtraia − 17 de − 17.

6-10. Repita o Problema 6-9 para os seguintes casos e mostre que ocorre overflow em cada caso.
(a) Some + 37 com + 95. **(b)** Subtraia + 37 de − 95.

SEÇÕES 6-5 E 6-6

6-11. Multiplique os seguintes pares de números binários e verifique seus resultados efetuando a multiplicação em decimal.

(a) 111 × 101
(b) 1011 × 1011
(c) 101,101 × 110,010
(d) 0,1101 × 0,1011

6-12. Realize as divisões a seguir. Verifique seus resultados fazendo a divisão em decimal.

(a) 1100 ÷ 100
(b) 111111 ÷ 1001
(c) 10111 ÷ 100
(d) 10110,1101 ÷ 1,1

SEÇÕES 6-7 E 6-8

6-13. Some os números decimais a seguir após convertê-los para seu código BCD.

(a) 74 + 23
(b) 58 + 37
(c) 147 + 380
(d) 385 + 118
(e) 998 + 003
(f) 623 + 599

6.14 Calcule a soma de cada um dos seguintes pares de números hexa.

(a) 3E91 + 2F93
(b) 91B + 6F2
(c) ABC + DEF
(d) 2FFE + 0002
(e) FFF + 0FF
(f) D191 + AAAB

6-15. Realize as seguintes subtrações com os pares de números hexa.

(a) 3E91 − 2F93
(b) 91B − 6F2
(c) 0300 − 005A
(d) 0200 − 0003
(e) F000 − EFFF
(f) 2F00 − 4000

6-16. O manual do usuário de um pequeno microcomputador afirma que o computador possui posições de memória disponíveis nos seguintes endereços hexa: 0200 até 03FF e de 4000 até 7FD0. Qual é o número total de posições de memória disponíveis?

6-17. **(a)** Uma determinada posição de memória tem um dado hexa 77. Se isto representar um número *sem sinal*, qual é o seu valor decimal?

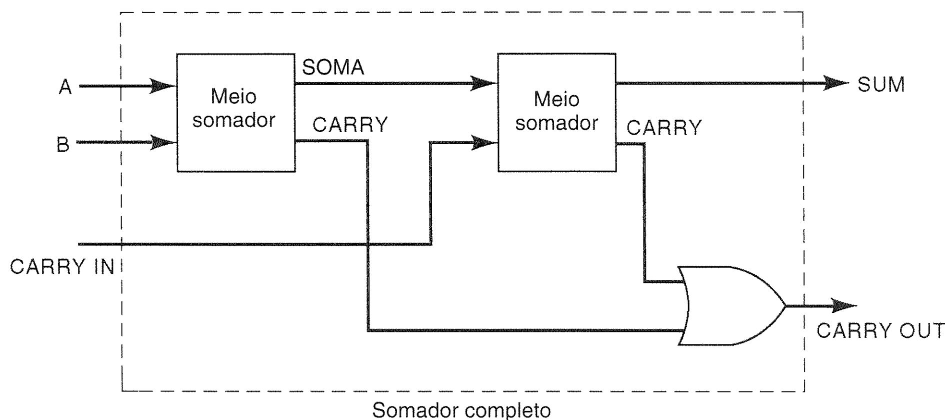


Fig. 6-20 - Problema 6-20.

(b) Se isto representar um número *com sinal*, qual é o seu valor decimal?
(c) Repita (a) e (b) se o valor do dado for E5.

SEÇÃO 6-11

6-18. Converta o circuito do somador completo da Fig. 6-7 para utilizar apenas portas NAND.

6-19. Escreva a tabela-verdade para um meio somador (entradas *A* e *B*; saídas SOMA e CARRY). A partir da tabela-verdade, projete um circuito lógico que funcionará com um meio somador.

6-20. Um somador completo pode ser implementado de muitas maneiras diferentes. A Fig. 6-20 mostra como ele pode ser construído com dois meios somadores. Construa a tabela-verdade para este arranjo e verifique que ele opera como um somador completo.

SEÇÃO 6-12

6-21. Veja a Fig. 6-9. Determine o conteúdo do registrador *A* após a seqüência de operações: [*A*] = 0000, [0100] → [*B*], [*S*] → [*A*], [1011] → [*B*], [*S*] → [*A*].

6-22. Veja a Fig. 6-9. Considere que cada FF tem $t_{PLH} = t_{PHL} = 30$ ns e um tempo de setup de 10 ns, e que cada somador completo tem um atraso de propagação de 40 ns. Qual é o tempo mínimo permitido entre a transição positiva do pulso de LOAD e a transição positiva do pulso de TRANSFER para uma operação adequada?

D

6-23. Nos circuitos somador e subtrator tratados neste capítulo, não levamos em consideração a possibilidade de *overflow*. O overflow ocorre quando os dois números sendo somados ou subtraídos produzem um resultado que contém mais bits do que a capacidade do acumulador. Por exemplo, utilizando-se registradores de quatro bits, incluindo o bit de sinal, os números que vão desde +7 até −8 (em complemento a 2) podem ser armazenados. Portanto, se o resultado de uma adição ou subtração exceder +7 ou −8, poderíamos dizer que ocorreu um overflow. Quando ocorre um overflow, os resultados são inúteis, pois eles não estão armazenados corretamente no registrador acumulador. Para ilustrar, some +5 (0101) e +4 (0100), o que resulta em 1001. Este 1001 poderia ser interpretado incorretamente como um número negativo, já que existe um 1 na posição do bit de sinal.

Em computadores e calculadoras normalmente existem circuitos que são usados para detectar uma condição de overflow. Existem várias maneiras de fazer isto. Um método que pode ser usado para o somador que opera em complemento a 2 funciona do seguinte modo:

1. Examine os bits de sinal dos números que estão sendo somados.
2. Examine o bit de sinal do resultado.
3. O overflow ocorre sempre que os números sendo somados são *ambos positivos* e o bit de sinal do resultado é 1 *ou* quando os números são *ambos negativos* e o bit de sinal do resultado é 0.

Este método pode ser verificado testando-se diversos exemplos. Os estudantes deveriam tentar os exemplos a seguir para maiores esclarecimentos: **(1)** 5 + 4; **(2)** −4 + (−6); **(3)** 3 + 2. Os casos 1 e 2 produzirão um overflow, e o caso 3 não. Logo, examinando-se os bits de sinal, pode-se projetar um circuito lógico que produzirá 1 na saída sempre que a condição de overflow ocorrer. Projete este circuito de overflow para o somador da Fig. 6-9.

C, D

6-24. Acrescente os circuitos lógicos necessários na Fig. 6-9 para acomodar a transferência de dados da memória para o registrador *A*. Os dados da memória devem entrar no registrador *A* através das suas entradas *D* na transição positiva do *primeiro* pulso TRANSFER; os dados das saídas de soma dos SCs serão carregados em *A* na transição positiva do *segundo* pulso TRANSFER. Em outras palavras, um pulso de LOAD seguido por dois pulsos TRANSFER são necessários para realizar a seqüência completa de carga do registrador *B* da memória, carga do registrador *A* da memória, e então transferir a soma para o registrador *A*. (*Sugestão:* Utilize um flip-flop *X* para controlar qual fonte de dados vai ser carregada nas entradas *D* do acumulador.)

SEÇÃO 6-13

C, D

6-25. Projete um circuito de carry antecipado para o somador da Fig. 6-9, que gera o carry C_3 a ser ligado ao SC da posição MSB, baseado nos valores de A_0, B_0, C_0, A_1, B_1, A_2 e B_2. Em outras palavras, derive a expressão para C_3 em termos de A_0, B_0, C_0, A_1, B_1, A_2 e B_2. (*Sugestão:* Comece escrevendo a expressão para C_1 em termos de A_0, B_0 e C_0. Então, escreva a expressão para C_2 em termos de A_1, B_1 e C_1. Substitua a expressão para C_1 na expressão para C_2. Então, escreva a expressão para C_3 em termos de A_2, B_2 e C_2. Substitua a

expressão para C_2 na expressão para C_3. Simplifique a expressão final para C_3 e coloque-a na forma de soma-de-produtos. Implemente o circuito.)

SEÇÃO 6-14

6-26. Mostre os níveis lógicos em cada entrada e saída da Fig. 6-10(a) quando 354_8 é somado com 103_8.

SEÇÃO 6-15

6-27. Para o circuito da Fig. 6-13, determine as saídas de soma para os seguintes casos.

(a) Registrador A = 0101 (+5), registrador B = 1110 (−2); SUB = 1, ADD = 0

(b) Registrador A = 1100 (−4), registrador B = 1110 (−2); SUB = 0, ADD = 1

(c) Repita o item (b) com ADD = SUB = 0.

D

6-28. Mostre como as portas da Fig. 6-13 podem ser implementadas usando três chips 74LS00.

D

6-29. Modifique o circuito da Fig. 6-13, de modo que uma única entrada de controle, X, seja usada no lugar de ADD e SUB. O circuito deve funcionar como um somador quando X = 0, e como um subtrator quando X = 1. A seguir, simplifique cada conjunto de portas. (*Sugestão:* Observe que agora cada conjunto de portas está funcionando como um inversor controlado.)

SEÇÃO 6-16

6-30. Suponha as seguintes entradas na Fig. 6-14: $[A]$ = 0101, $[B]$ = 1001, C_0 = 0. Determine os níveis lógicos de $[S]$, X, $[\Sigma]$ e do CARRY.

C

6-31. Faria alguma diferença no somador BCD da Fig. 6-14 se o C_0 do somador superior fosse mantido em BAIXO enquanto o C_0 do somador inferior fosse usado como a entrada de carry? Explique.

6-32. Suponha que o registrador A na Fig. 6-15 contém o código BCD de 376 e que o registrador B contém o código BCD para 469. Determine as saídas.

SEÇÃO 6-17

6-33. Determine as saídas F, C_{N+4} e OVR para cada um dos seguintes conjuntos de entradas aplicadas em um 74LS382.

(a) $[S]$ = 011, $[A]$ = 0110, $[B]$ = 0011, $C_N = 0$
(b) $[S]$ = 001, $[A]$ = 0110, $[B]$ = 0011, $C_N = 1$
(c) $[S]$ = 010, $[A]$ = 0110, $[B]$ = 0011, $C_N = 1$

D

6-34. Mostre como o 74HC382 pode ser usado para produzir $[F] = [\overline{A}]$. (*Sugestão:* Lembre-se daquela propriedade especial de uma porta EX-OR.)

6-35. Determine as saídas Σ na Fig. 6-17 para os seguintes conjuntos de entradas.

(a) $[S]$ = 110, $[A]$ = 10101100, $[B]$ = 00001111
(b) $[S]$ = 100, $[A]$ = 11101110, $[B]$ = 00110010

C, D

6-36. Acrescente a lógica necessária na Fig. 6-17 para produzir uma única saída em ALTO sempre que o número binário em A for exatamente o mesmo que o número binário em B. Aplique o código de seleção de entrada apropriado (três códigos podem ser usados).

SEÇÃO 6-19

T

6-37. Considere o circuito da Fig. 6-9. Suponha que a saída A_2 está permanentemente em nível BAIXO. Siga a seqüência de operações para somar dois números e determine os resultados que aparecerão no registrador A após o segundo pulso TRANSFER, para cada um dos seguintes casos. Note que os números estão em decimal e que o primeiro número é carregado em B pelo primeiro pulso de LOAD.

(a) 2 + 3
(b) 3 + 7
(c) 7 + 3
(d) 8 + 3
(e) 9 + 3

T

6-38. Uma estudante montou o circuito do somador/subtrator da Fig. 6-13. Durante os testes, ela constata que sempre que uma adição é realizada o resultado é 1 unidade a mais do que o esperado, e quando uma subtração é realizada o resultado é 1 unidade a menos do que o esperado. Qual é o erro provável que a estudante cometeu ao montar o circuito?

T

6-39. O somador BCD da Fig. 6-14 é testado, e os resultados estão registrados na Tabela 6-4. Considere cada uma das possíveis falhas e indique se ela poderia ou não ser a falha real. Explique cada resposta.

(a) As entradas A_1 e A_0 do somador de correção estão internamente em curto-circuito.

(b) Existe alguma ligação interrompida de X para o somador de correção.

(c) As entradas da porta OR superior estão internamente em curto-circuito.

(d) A saída da porta AND está permanentemente em BAIXO.

TABELA 6-4

	$B_3B_2B_1B_0$	$A_3A_2A_1A_0$	$\Sigma_3\Sigma_2\Sigma_1\Sigma_0$	CARRY (X)
(1)	0011	0110	1001	0
(2)	0111	1000	1111	0
(3)	1001	1001	0010	0

QUESTÃO DE FIXAÇÃO

6-40. Defina cada um dos seguintes termos.

(a) Somador completo
(b) Complemento a 2
(c) Unidade lógica e aritmética
(d) Bit de sinal
(e) Overflow
(f) Acumulador
(g) Somador paralelo
(h) Carry antecipado
(i) Negação
(j) Registrador B

APLICAÇÕES EM MICROCOMPUTADORES

C, D

6-41. Em uma ULA típica de microprocessador, os resultados de todas as operações aritméticas são usualmente (mas nem sempre) transferidos para o registrador acumulador como nas Figs. 6-9, 6-13 e 6-16. Na maioria das ULAs de

microprocessadores, o resultado de cada operação aritmética também é usado para controlar os estados de vários flip-flops especiais denominados *flags*. Estes flags são usados pelo microprocessador quando ele está tomando decisões durante a execução de certos tipos de instrução. Os três flags mais comuns são:

S (flag de sinal). Este FF tem sempre o mesmo estado que o sinal do último resultado da ULA.
Z (flag de zero). Este flag vai para 1 sempre que o resultado da operação na ULA é 0. Caso contrário, o flag é limpo para 0.
C (flag de carry). Este FF é sempre igual ao carry do MSB da ULA.

Utilizando o somador/subtrator da Fig. 6-13 como se fosse uma ULA, projete o circuito lógico que implementa estes flags. As saídas da soma e a saída C_4 são usadas para controlar para que estado cada flag irá quando da ocorrência do pulso TRANSFER. Por exemplo, se a soma é exatamente 0 (isto é, 0000), o flag Z deveria ser setado pela transição positiva de TRANSFER; caso contrário, ele deveria ser limpo.

6-42. No trabalho com microcomputadores, freqüentemente é necessário mover números binários de um registrador de 8 bits para um registrador de 16 bits. Considere os números 01001001 e 10101110, que representam +73 e −82, respectivamente, no sistema de complemento a 2. Determine as representações de 16 bits para esses números decimais.

6-43. Compare as representações de 8 e 16 bits para +73 do Problema 6-42. Então, compare as duas representações para −82. Existe uma regra geral que pode ser usada para converter facilmente da representação de 8 bits para a de 16 bits. Você pode perceber qual é? Ela tem algo relacionado ao bit de sinal do número de 8 bits.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 6-1
1. **(a)** 11101 **(b)** 101,111 **(c)** 10010000

SEÇÃO 6-2
1. **(a)** 00001101 **(b)** 11111001 **(c)** 10000000
2. **(a)** −29 **(b)** −64 **(c)** +126
3. −2.048 até +2.047
4. Sete **5.** −32.768
6. **(a)** 10000 **(b)** 10000000 **(c)** 1000
7. Vide texto.

SEÇÃO 6-3
1. Verdadeiro 2. **(a)** $100010_2 = -30_{10}$ **(b)** $000000_2 = 0_{10}$

SEÇÃO 6-4
1. **(a)** $01111_2 = +15_{10}$ **(b)** $11111_2 = -1_{10}$
2. Comparando-se o bit de sinal da soma com os bits de sinal dos números sendo somados

SEÇÃO 6-5
1. 1100010

SEÇÃO 6-7
1. A soma em pelo menos uma posição decimal é maior do que 1001 (9).
2. O fator de correção é adicionado nas posições das unidades e das dezenas.

SEÇÃO 6-8
1. 923 **2.** 3DB **3.** 2F, 77EC, 6D

SEÇÃO 6-10
1. Três; dois **2.** **(a)** $S_2 = 0$, $C_3 = 1$ **(b)** $C_5 = 0$

SEÇÃO 6-12
1. Um; quatro; quatro 2. 0100

SEÇÃO 6-14
1. Cinco chips 2. 240 ns

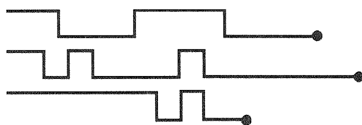
SEÇÃO 6-15
1. Para somar o 1 necessário para completar a representação em complemento a 2 do número no registrador *B*
2. 0010 **3.** 1101
4. Falso; o complemento a 1 aparece lá.

SEÇÃO 6-16
1. Dois somadores de quatro bits e a lógica de correção
2. A lógica de correção detecta uma soma maior do que 9 e então faz com que 0110 seja adicionado à soma.

SEÇÃO 6-17
1. $F = 1011$; $OVR = 0$; $C_{N+4} = 0$
2. $F = 0111$; $OVR = 1$; $C_{N+4} = 1$
3. $F = 1000$
4. $\Sigma = 01101011$; $C_{N+4} = OVR = 0$
5. $\Sigma = 11111111$
6. Oito

CAPÍTULO 7

Contadores e Registradores



■ SUMÁRIO

OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Compreender a operação e as características dos contadores síncronos e assíncronos.
- Construir contadores com módulo menor do que 2^N.
- Identificar os símbolos IEEE/ANSI utilizados para CIs contadores e registradores.
- Construir contadores tanto crescentes quanto decrescentes.
- Conectar contadores de vários estágios.
- Analisar e avaliar os vários tipos de contadores com carga paralela.
- Projetar contadores síncronos de seqüências arbitrárias.
- Entender diversas formas usadas para decodificar diferentes tipos de contadores.
- Antecipar e eliminar os efeitos dos glitches de decodificação.
- Comparar as principais diferenças entre contadores em anel e contadores Johnson.
- Analisar a operação de um freqüencímetro e de um relógio digital.
- Reconhecer e compreender a operação de vários tipos de CIs de registradores.
- Aplicar as técnicas de pesquisa de falhas usadas para sistemas lógicos combinacionais para depurar sistemas lógicos seqüenciais.

INTRODUÇÃO

No Cap. 5 vimos como os flip-flops podem ser conectados para operar como contadores e registradores. Até agora estudamos apenas os circuitos básicos de contadores e registradores. Sistemas digitais empregam diversas variações desses circuitos básicos, em sua maioria sob a forma de circuitos integrados. Neste capítulo analisaremos como FFs e portas lógicas podem ser combinados para produzir tipos diferentes de contadores e registradores.

Tendo em vista que existe um grande número de tópicos neste capítulo, ele foi dividido em duas partes. Na **PARTE I** abordaremos os princípios de operação dos contadores, os vários arranjos de circuitos contadores e CIs contadores representativos. Na **PARTE II** apresentaremos as aplicações de contadores, tipos de CIs registradores e pesquisa de falhas.

Compreender o material deste capítulo é uma boa indicação de que o material dos capítulos anteriores foi aprendido.

PARTE I

7-1 CONTADORES ASSÍNCRONOS

A Fig. 7-1 mostra um contador binário de quatro bits similar ao que foi apresentado no Cap. 5. Lembre-se dos seguintes pontos relativos à sua operação:

1. Os pulsos de clock são aplicados somente na entrada *CLK* do flip-flop *A*. Deste modo, o flip-flop *A* comuta (vai para o seu estado oposto) toda vez que os pulsos de clock têm uma transição negativa (de ALTO para BAIXO). Note que $J = K = 1$ para todos os FFs.
2. A saída normal do flip-flop *A* aciona a entrada *CLK* do flip-flop *B*, e portanto o flip-flop *B* comutará sempre que a saída *A* for de 1 para 0. Analogamente, o flip-flop *C* comutará quando *B* for de 1 para 0, e o flip-flop *D* comutará quando *C* for de 1 para 0.
3. As saídas *D*, *C*, *B* e *A* dos FFs representam um número binário de quatro bits, com *D* sendo o MSB. Vamos considerar que todos os FFs foram limpos para o estado 0 (as entradas CLEAR não estão mostradas). As formas de onda na Fig. 7-1 mostram que uma seqüência de contagem binária de 0000 até 1111 é gerada conforme os pulsos de clock são continuamente aplicados.
4. Após a descida do décimo quinto pulso de clock ter ocorrido, os FFs do contador estão com 1111. Na décima sexta descida, o flip-flop *A* vai de 1 para 0, o que faz com que o flip-flop *B* vá de 1 para 0, e assim por diante até que o contador alcance o estado 0000. Em outras palavras, o contador realizou um ciclo completo (0000 até 1111) e *reciclou* de volta para 0000, de onde reiniciará um novo ciclo de contagem conforme pulsos de clock forem aplicados.

Neste contador, cada saída de FF aciona a entrada *CLK* do próximo FF. Este tipo de arranjo para contador é denominado **contador assíncrono** porque os FFs não trocam de estado em exato sincronismo com os pulsos de clock aplicados; apenas o flip-flop *A* responde aos pulsos de clock. O FF *B* deve esperar o FF *A* mudar de estado antes de poder comutar; o FF *C* deve aguardar pelo FF *B*, e assim por diante. Assim, existe um atraso entre as respostas dos sucessivos FFs. Este atraso é tipicamente de 5-20 ns por FF. Em alguns casos, como veremos, este atraso pode ser problemático. Este tipo de contador é também freqüentemente chamado de **contador por pulsação** (*ripple counter*) devido ao modo como os FFs respondem um após o outro numa espécie de efeito análogo à propagação de um pulso. Utilizaremos de modo intercambiável os termos *contador assíncrono* e *contador por pulsação.*

Fluxo do Sinal

Em diagramas esquemáticos, é convenção desenhar os circuitos (sempre que possível) de modo que o fluxo do sinal seja da esquerda para a direita, com as entradas na esquerda e as saídas na direita. Neste capítulo freqüentemente quebraremos esta convenção, especialmente em diagramas que mostram contadores. Por exemplo, na Fig. 7-1, as entradas *CLK* de cada FF estão na direita, as saídas estão na

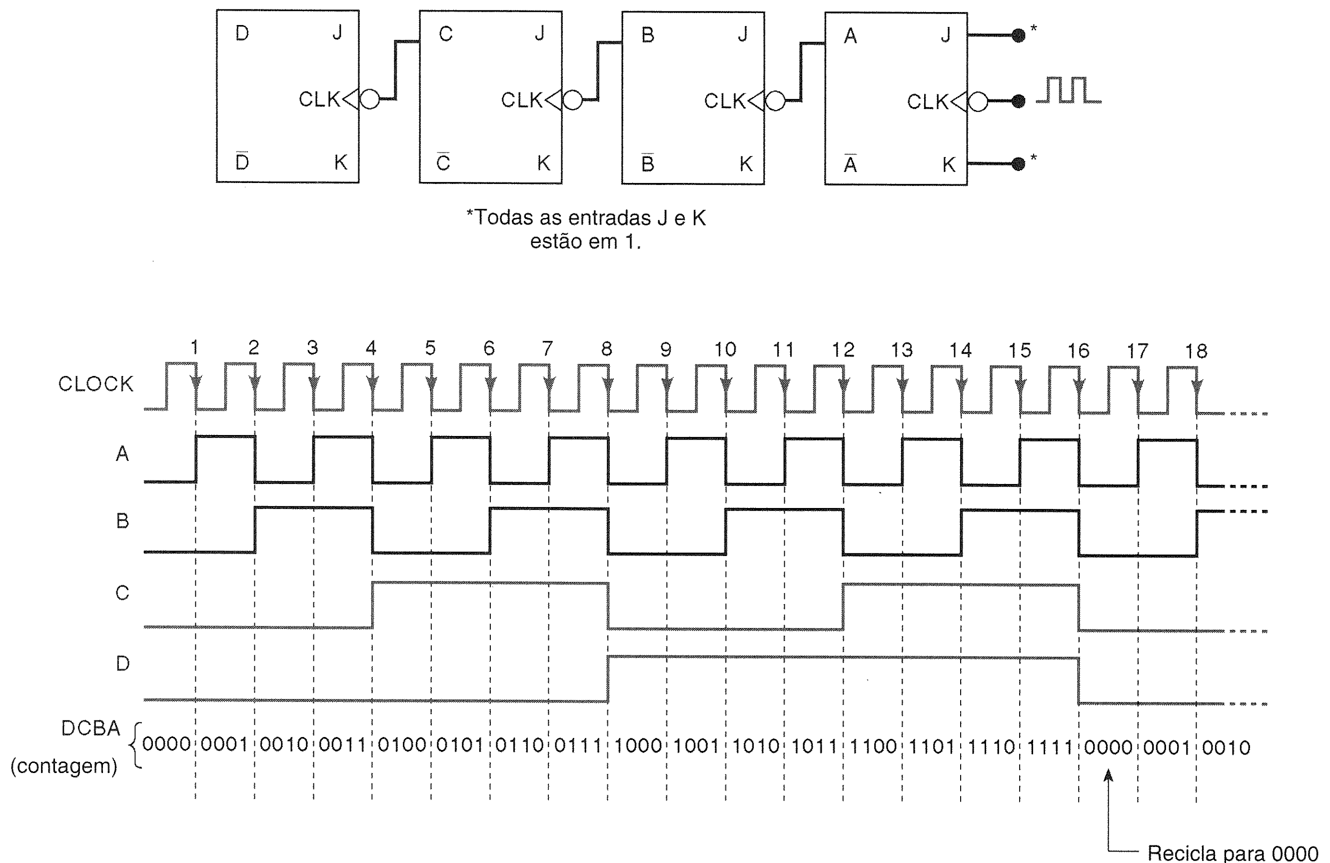


Fig. 7-1 - Contador assíncrono de quatro bits.

esquerda e o sinal de clock de entrada é mostrado vindo da direita. Utilizaremos este arranjo porque ele faz a operação do contador mais fácil de ser compreendida e acompanhada, já que a ordem dos FFs é a mesma que a ordem dos bits no número binário que o contador representa. Em outras palavras, o FF *A*, que é o LSB, é o FF mais à direita, e o FF *D*, que é o MSB, é o mais à esquerda. Se utilizássemos o fluxo de sinal convencional da esquerda para a direita, teríamos que colocar o FF *A* na esquerda e o FF *D* na direita, que é o oposto das posições no número binário que o contador representa. Em alguns dos diagramas com contadores deste capítulo, empregaremos o fluxo de sinal convencional da esquerda para a direita para você se acostumar a vê-lo.

EXEMPLO 7-1

O contador da Fig. 7-1 começa no estado 0000, e então pulsos de clock são aplicados. Algum tempo depois, os pulsos de clock são removidos, e os FFs do contador apresentam 0011. Quantos pulsos de clock ocorreram?

Solução

A resposta parece ser 3, já que 0011 é o equivalente binário de 3. Entretanto, com as informações dadas, não há como saber se o contador reciclou ou não. Isto significa que 19 pulsos de clock poderiam ter ocorrido; os primeiros 16 teriam trazido o contador de volta a 0000, e os 3 últimos o teriam levado para 0011. Poderia ter havido 35 pulsos (dois ciclos completos e então mais três), ou 51 pulsos, e assim por diante.

Módulo

O contador na Fig. 7-1 tem 16 estados distintos (0000 até 1111). Logo, ele é um *contador assíncrono de módulo 16*. Lembre-se de que o **módulo** é sempre igual ao número de estados que o contador atinge em cada ciclo completo antes de reciclar de volta ao estado inicial. O módulo pode ser aumentado simplesmente adicionando-se mais FFs ao contador. Isto é,

$$\text{Módulo} = 2^N \qquad (7\text{-}1)$$

onde N é o número de FFs conectados segundo o arranjo da Fig. 7-1.

EXEMPLO 7-2

Um contador é necessário para contar o número de itens que passam sobre uma esteira de transporte. Uma combinação de uma fonte de luz com uma fotocélula é usada para gerar um único pulso cada vez que um item cruza a trajetória do feixe. O contador deve ser capaz de contar mil itens. Quantos FFs são necessários?

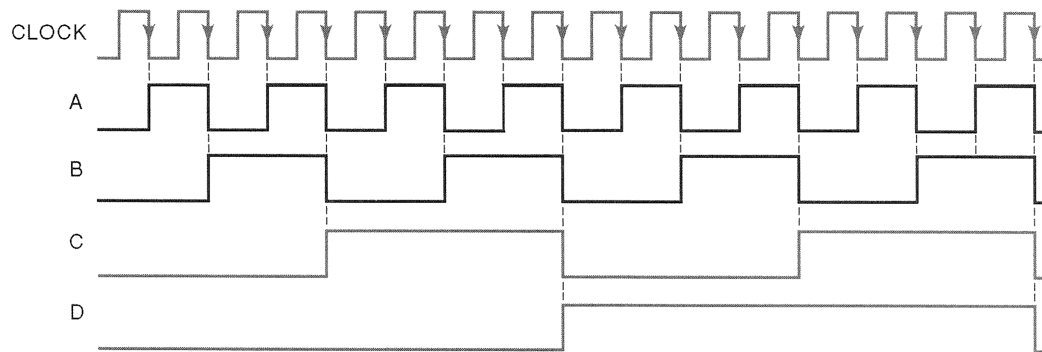


Fig. 7-2 - Formas de onda do contador mostrando a divisão de freqüência por 2 em cada FF.

Solução
Simplesmente basta determinar qual valor de N é necessário para que $2^N \geq 1000$. Já que $2^9 = 512$, 9 FFs não serão suficientes. $2^{10} = 1024$, portanto 10 FFs produziriam um contador que poderia contar até $1111111111_2 = 1023_{10}$. Assim, deveríamos usar 10 FFs. Poderíamos usar mais do que 10, mas seria um desperdício de FFs, já que qualquer FF além do décimo não seria necessário.

Divisão de Freqüência

No Cap. 5, vimos que no contador básico cada FF fornece uma forma de onda de saída que tem exatamente a *metade* da freqüência da forma de onda em sua entrada *CLK*. Para ilustrar, suponha que o sinal de clock na Fig. 7-1 tem 16 kHz. A Fig. 7-2 mostra as formas de onda de saída dos FFs. A forma de onda da saída *A* é uma *onda quadrada* de 8 kHz, na saída *B* é de 4 kHz, na saída *C* é de 2 kHz e na saída *D* é de 1 kHz. Note que a saída do flip-flop *D* tem uma freqüência igual à freqüência original do clock dividida por 16. De um modo geral, *para qualquer contador, a saída do último FF (isto é, do MSB) divide a freqüência do clock de entrada pelo módulo do contador.* Por exemplo, um contador de módulo 16 poderia também ser chamado de *contador divisor por 16.*

EXEMPLO 7-3

O primeiro passo para a construção de um relógio digital é acionar um circuito Schmitt-trigger conformador de pulsos* para produzir uma onda quadrada conforme ilustra a Fig. 7-3. A onda quadrada de 60 Hz é levada para um contador de módulo 60, que é usado para dividir a freqüência de 60 Hz exatamente por 60, para produzir uma forma de onda de 1 Hz. Essa forma de onda de 1 Hz é levada para uma série de contadores, que, então, contam os segundos, minutos, horas e assim por diante. Quantos FFs são necessários para o contador de módulo 60?

Solução
Não existe nenhuma potência inteira de 2 que seja igual a 60. A mais próxima é $2^6 = 64$. Assim, um contador utilizando seis FFs funcionaria como um contador de módulo 64. Obviamente, isto não satisfará o requisito. Parece que não há solução usando um contador do tipo mostrado na Fig. 7-1. Isso está parcialmente correto; na próxima seção veremos como modificar este contador binário básico, de modo que praticamente *qualquer* módulo possa ser obtido e não estaremos limitados a valores de 2^N.

Questões de Revisão

1. *Verdadeiro ou falso*: Em um contador assíncrono, todos os FFs trocam de estado ao mesmo tempo.
2. Considere que o contador na Fig. 7-1 está com a contagem de 0101. Qual será a contagem após 27 pulsos de clock?
3. Qual seria o módulo do contador se três FFs fossem adicionados?

7-2 CONTADORES DE MÓDULO $< 2^N$

O contador por pulsação básico da Fig. 7-1 está limitado a módulos que são iguais a 2^N, onde N é o número de FFs. Esse valor, na verdade, é o módulo máximo que pode ser obtido utilizando N flip-flops. O contador básico pode ser

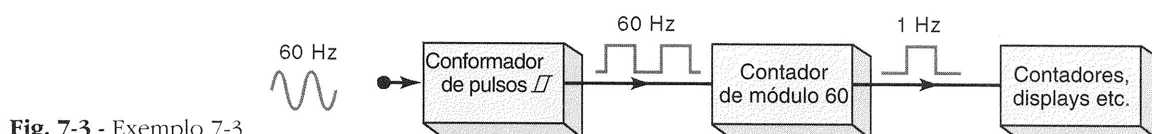


Fig. 7-3 - Exemplo 7-3.

*Vide Seção 5-21.

modificado para produzir módulos menores do que 2^N, permitindo que o contador *pule estados* que normalmente fariam parte da seqüência de contagem. Um dos métodos mais comuns para realizar isto está ilustrado na Fig. 7-4, onde um contador por pulsação de três bits é mostrado. Ignorando a porta NAND por um momento, podemos ver que o contador é um contador binário de módulo 8 que contará em seqüência de 000 a 111. Entretanto, a presença da porta NAND alterará essa seqüência como segue:

1. A saída da NAND está conectada nas entradas assíncronas de CLEAR dos FFs. Enquanto a saída da NAND estiver em ALTO, ela não terá efeito sobre o contador. Quando ela vai para BAIXO, no entanto, ela vai limpar todos os FFs, de modo que o contador imediatamente vai para o estado 000.

2. As entradas da porta NAND são as saídas dos flip-flops *B* e *C*, e, portanto, a saída da NAND vai para BAIXO sempre que $B = C = 1$. Esta condição ocorrerá quando o contador for do estado 101 para o estado 110, na descida do pulso de entrada 6. O nível BAIXO na saída da NAND limpará imediatamente (geralmente dentro de uns poucos nanossegundos) o contador para o estado 000. Uma vez que os FFs tenham sido limpos, a saída da NAND retorna para ALTO, já que a condição $B = C = 1$ não existe mais.

3. A seqüência de contagem é, portanto,

CBA
000 ←
001
010
011
100
101
110 → (estado temporário necessário para limpar o contador)

Embora o contador vá para o estado 110, ele permanece lá por apenas alguns nanossegundos antes de reciclar para 000. Assim, podemos dizer que esse contador conta de 000 (zero) até 101 (cinco) e então recicla para 000. Ele essencialmente pulou 110 e 111, de modo que ele tem apenas seis estados diferentes; logo, ele é um contador de módulo 6.

Note que a forma de onda da saída *B* contém um *spike* ou *glitch* causado pela ocorrência momentânea do estado 110 antes de o contador ser limpo. Este glitch é muito estreito, e portanto não produziria qualquer indicação visual em LEDs ou displays numéricos. Ele poderia, entretanto, causar um problema se a saída *B* estivesse sendo usada para acionar outros circuitos externos ao contador. Também

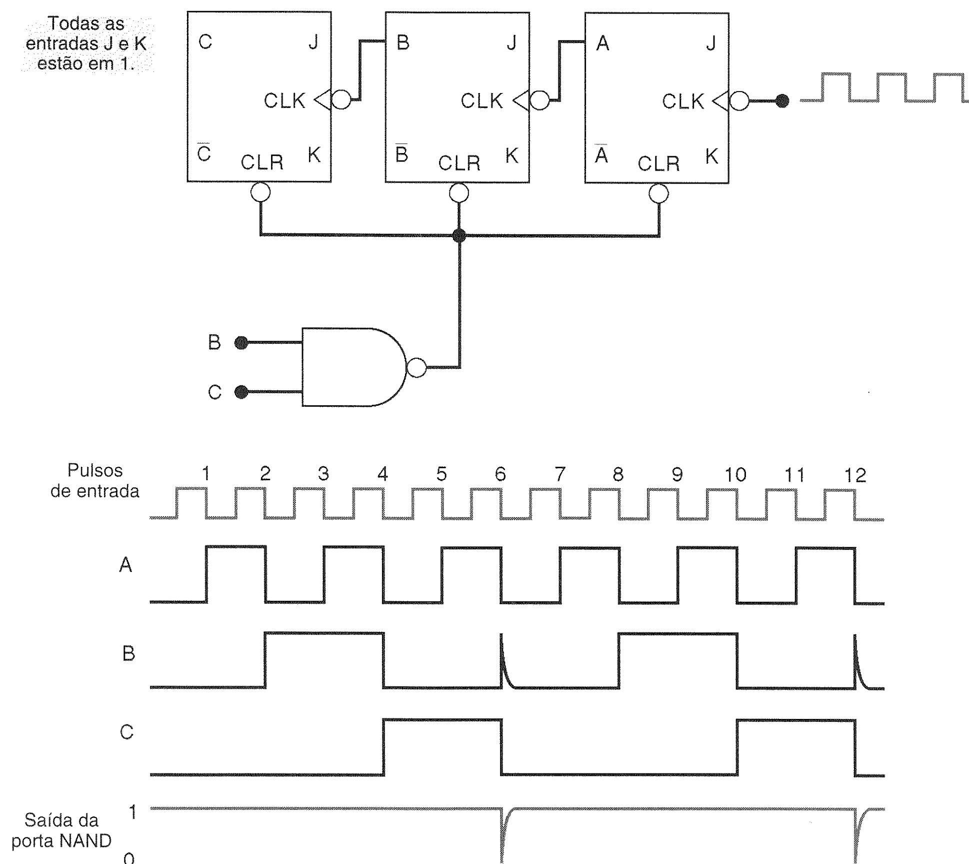


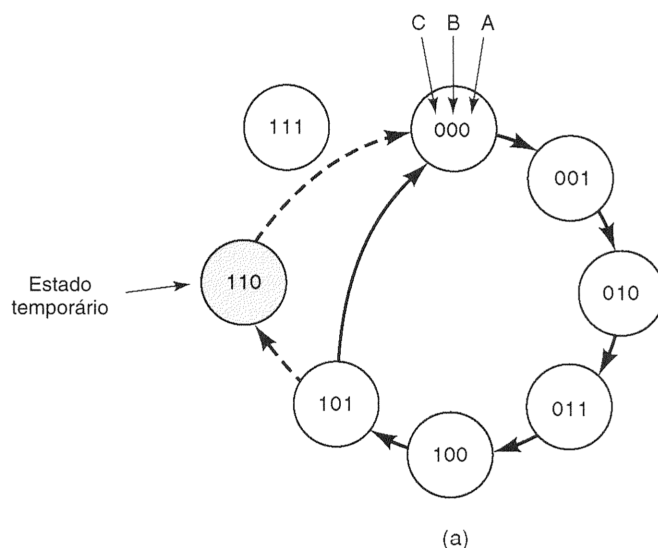
Fig. 7-4 - Contador de módulo 6 obtido pelo acionamento do CLEAR de um contador de módulo 8 quando a contagem seis (110) ocorre.

deveria ser notado que a saída *C* tem uma freqüência igual a um sexto da freqüência de entrada; em outras palavras, esse contador de módulo 6 dividiu a freqüência de entrada por *seis*. A forma de onda de *C não* é uma onda quadrada simétrica (50% de taxa de ciclo), pois está em ALTO por apenas dois pulsos de clock e em BAIXO por quatro ciclos.

Diagrama de Transição de Estados

A Fig. 7-5(a) é o diagrama de transição de estados para o contador de módulo 6 da Fig. 7-4, que mostra como os FFs *C*, *B* e *A* mudam de estado conforme os pulsos são aplicados na entrada *CLK* do flip-flop *A*. Lembre-se de que cada círculo representa um dos possíveis estados do contador e de que as setas indicam como ocorre uma mudança de estado em resposta a um pulso de clock de entrada.

Se presumimos uma contagem inicial de 000, o diagrama mostra que os estados do contador se alteram normalmente, para cima, até uma contagem de 101. Quando o próximo pulso de clock ocorre, o contador vai temporariamente para a contagem 110, antes de ir para a contagem estável de 000. As linhas tracejadas indicam a natureza temporária do estado 110. Conforme dito anteriormente, a duração deste estado temporário é tão curta que para a maio-



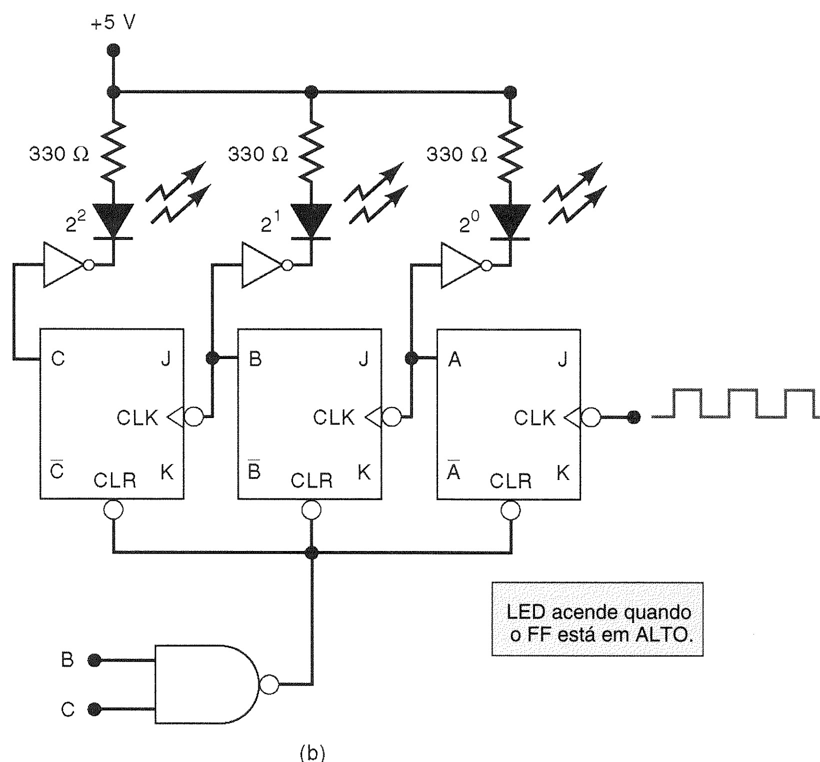


Fig. 7-5 - (a) Diagrama de transição de estados para o contador de módulo 6 da Fig. 7-4. (b) LEDs são utilizados freqüentemente para indicar os estados de um contador.

ria dos propósitos podemos considerar que o contador vai diretamente de 101 para 000 (linha cheia).

Note que o estado 111 jamais é alcançado, nem mesmo temporariamente.

Mostrando os Estados do Contador

Algumas vezes durante a operação normal, e muito freqüentemente durante os testes, é necessário obter uma informação visual de como um contador está mudando de estado em resposta aos pulsos de entrada. Vamos analisar detalhadamente diversos modos de fazer isto mais adiante no livro. Por ora, a Fig. 7-5(b) mostra um dos métodos mais simples que usa LEDs indicadores individuais para cada saída de FF. Cada uma das saídas dos FFs é conectada num INVERSOR cuja saída fornece o caminho para a corrente do LED. Por exemplo, quando a saída *A* está em ALTO, a saída do INVERSOR vai para BAIXO e o LED acende. O LED aceso indica $A = 1$. Quando a saída *A* está em BAIXO, a saída do INVERSOR está em ALTO e o LED apaga. O LED apagado indica $A = 0$.

EXEMPLO 7-4

(a) Qual é o estado dos LEDs quando o contador apresenta uma contagem de cinco?
(b) O que os LEDs mostram quando o contador é acionado por uma entrada de 1 kHz?
(c) O estado 110 será visível nos LEDs?

Solução

(a) Já que $5_{10} = 101_2$, os LEDs de 2^0 e 2^2 estarão acesos e o LED de 2^1 estará apagado.
(b) A 1 kHz, os LEDs estarão comutando entre ligado e desligado tão rapidamente que eles parecerão acesos para o olho humano com aproximadamente metade do brilho normal.
(c) Não; o estado 110 existirá por apenas alguns nanossegundos quando o contador retorna para 000.

Alterando o Módulo

O contador das Figs. 7-4 e 7-5 é um contador de módulo 6 por causa da escolha das entradas da porta NAND. Qualquer módulo desejado pode ser obtido alterando estas entradas. Por exemplo, usando uma porta NAND de três entradas com entradas *A*, *B* e *C*, o contador funcionaria normalmente até a condição 111 ser alcançada; neste ponto, ele imediatamente seria ressetado para o estado 000. Ignorando a excursão temporária no estado 111, o contador iria de 000 até 110 e então retornaria de volta a 000, resultando em um contador de módulo 7 (sete estados).

EXEMPLO 7-5

Determine o módulo do contador da Fig. 7-6(a). Determine também a freqüência na saída *D*.

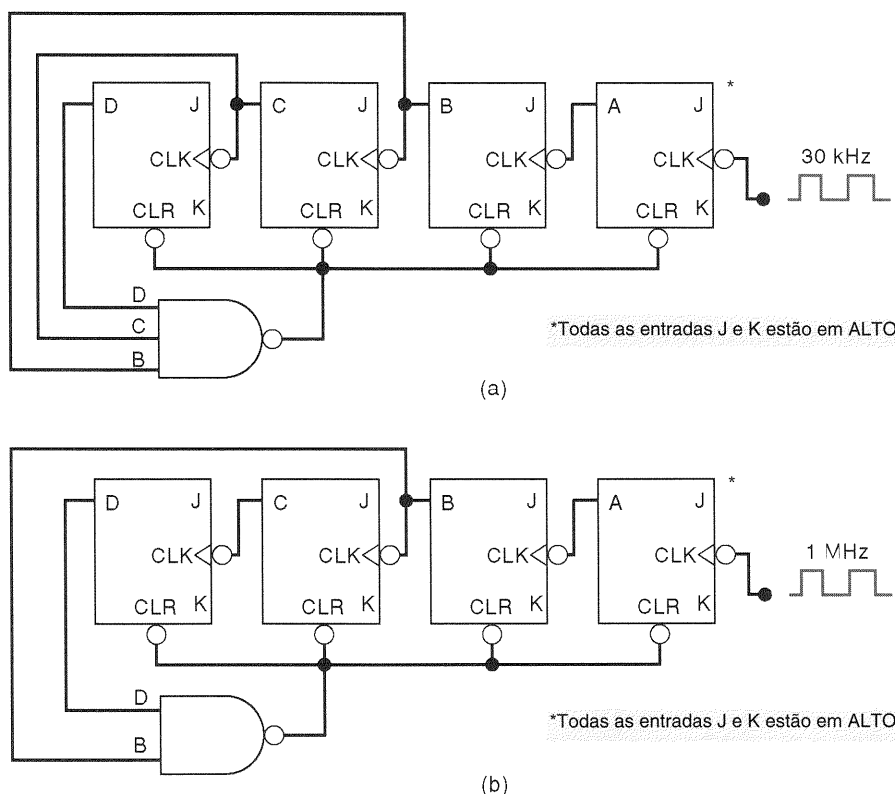


Fig. 7-6 - (a) Contador assíncrono de módulo 14; (b) contador assíncrono de módulo 10 (decádico).

Solução
Este é um contador de quatro bits, que normalmente contaria de 0000 até 1111. As entradas da NAND são *D*, *C* e *B*, o que significa que o contador reciclará imediatamente para 0000 quando a contagem 1110 (14 em decimal) for alcançada. Assim, o contador tem realmente 14 estados estáveis de 0000 até 1101 e é portanto um contador de *módulo 14*. Como a freqüência de entrada é de 30 kHz, a freqüência na saída *D* será

$$\frac{30 \text{ kHz}}{14} = 2{,}14 \text{ kHz}$$

Procedimento Geral

Para construir um contador que começa contando de 0 e tem módulo *X*:

1. Determine o menor número de FFs tal que $2^N \geq X$, e conecte-os como um contador. Se $2^N = X$, não faça os passos 2 e 3.
2. Conecte uma porta NAND nas entradas assíncronas de CLEAR de todos os FFs.
3. Determine quais FFs estarão no estado ALTO na contagem *X*; então conecte as saídas normais destes FFs nas entradas da porta NAND.

EXEMPLO 7-6

Construa um contador de módulo 10 que contará de 0000 (zero) até 1001 (9 decimal).

Solução
$2^3 = 8$ e $2^4 = 16$; logo, quatro FFs são necessários. Como o contador deve ter operação estável até a contagem de 1001, ele deve ser limpado para zero quando a contagem 1010 for alcançada. Portanto, as saídas dos FFs *D* e *B* devem ser conectadas nas entradas da porta NAND. A Fig. 7-6(b) mostra este arranjo.

Contadores Decádicos/Contadores BCD

O contador de módulo 10 do Exemplo 7-6 também é denominado **contador decádico**. De fato, um contador decádico é qualquer contador com 10 estados distintos, não importando em que seqüência. Um contador decádico como o da Fig. 7-6(b), que conta em seqüência desde 0000 (zero) até 1001 (9 decimal), é também comumente denominado **contador BCD**, pois usa apenas os 10 códigos BCD 0000, 0001, ..., 1000 e 1001. Para enfatizar, qualquer contador de módulo 10 é um contador decádico, e qualquer contador decádico que conta em binário desde 0000 até 1001 é um contador BCD.

Contadores decádicos, especialmente os do tipo BCD, encontram vasto uso em aplicações onde pulsos ou eventos devem ser contados e os resultados apresentados em algum tipo de mostrador numérico decimal. Examinaremos isto mais adiante com maiores detalhes. Um contador decádico também é bastante usado para dividir uma freqüência de pulsos *exatamente* por 10. Os pulsos de entrada são aplicados no flip-flop *A*, e os pulsos de saída são tomados da saída do flip-flop *D*, que tem um décimo da freqüência da entrada.

EXEMPLO 7-7

No Exemplo 7-3, um contador de módulo 60 foi necessário para dividir a freqüência de 60 Hz da rede para 1 Hz. Construa um contador de módulo 60.

Solução
$2^5 = 32$ e $2^6 = 64$, e portanto precisamos de seis FFs, conforme mostrado na Fig. 7-7. O contador deve ser limpado quando ele alcança a contagem de 60 (111100). Logo, as saídas dos flip-flops Q_2, Q_3, Q_4 e Q_5 devem ser conectadas na porta NAND. A saída do flip-flop Q_5 terá uma freqüência de 1 Hz.

Questões de Revisão

1. Quais as saídas dos FFs que devem ser conectadas na porta NAND para formar um contador de módulo 13?
2. *Verdadeiro ou falso:* Todos os contadores BCD são contadores decádicos.
3. Qual é a freqüência de saída de um contador decádico que é acionado por um sinal de 50 kHz?

7-3 CIRCUITOS INTEGRADOS DE CONTADORES ASSÍNCRONOS

Existem vários CIs TTL e CMOS de contadores assíncronos. Um deles é o TTL 74LS293. A Fig. 7-8(a) mostra o diagrama

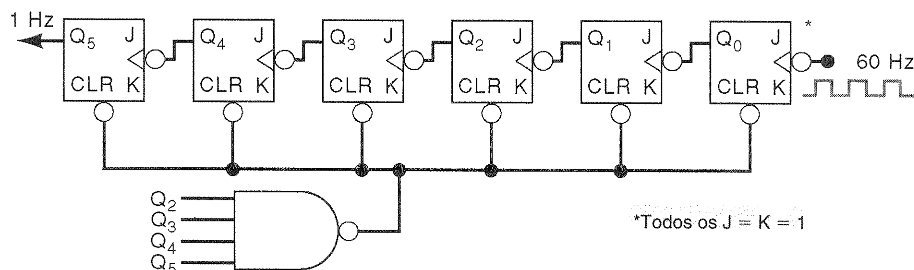


Fig. 7-7 - Contador de módulo 60.

lógico do 74LS293 conforme apareceria no manual TTL do fabricante. Alguns nomes são diferentes do que temos usado, mas é fácil compreendê-los. Note os seguintes pontos:

1. O 74LS293 tem quatro flip-flops J-K com saídas Q_0, Q_1, Q_2 e Q_3, onde Q_0 é o LSB e Q_3 é o MSB. Os FFs estão mostrados com o LSB à esquerda. Isto é feito para satisfazer a convenção de que os sinais de entrada em um circuito aparecem na esquerda. Temos desenhado nossos contadores com o LSB na direita, de modo que a ordem dos FFs é a mesma ordem dos bits na contagem binária.
2. Cada FF tem uma entrada *CP* (pulso de clock), que é apenas um outro nome para a entrada *CLK*. As entradas de clock para Q_0 e Q_1, identificadas como $\overline{CP}_0$ e $\overline{CP}_1$, respectivamente, são externamente acessíveis. As barras de inversão sobre estas entradas indicam que elas são ativadas na descida.
3. Cada FF tem uma entrada assíncrona de CLEAR, C_D. Elas estão conectadas juntas na saída de uma porta NAND de duas entradas, MR_1 e MR_2, onde *MR* significa *master reset* (reset geral). Ambas as entradas *MR* devem estar em ALTO para limpar o contador para 0000.
4. Os flip-flops Q_1, Q_2 e Q_3 já estão conectados como um contador por pulsação de três bits. O flip-flop Q_0 não é conectado internamente a nada. Isto permite ao usuário a opção de conectar Q_0 e Q_1 para formar um contador de quatro bits, ou usar Q_0 separadamente se desejar.

Os exemplos a seguir ilustrarão algumas maneiras de ligar o 74LS293 para produzir contadores diferentes. Nestes exemplos usaremos o símbolo lógico simplificado mostrado na Fig. 7-8(b).

EXEMPLO 7-8

Mostre como o 74LS293 poderia ser conectado para operar como um contador de módulo 16 com uma entrada de clock de 10 kHz.

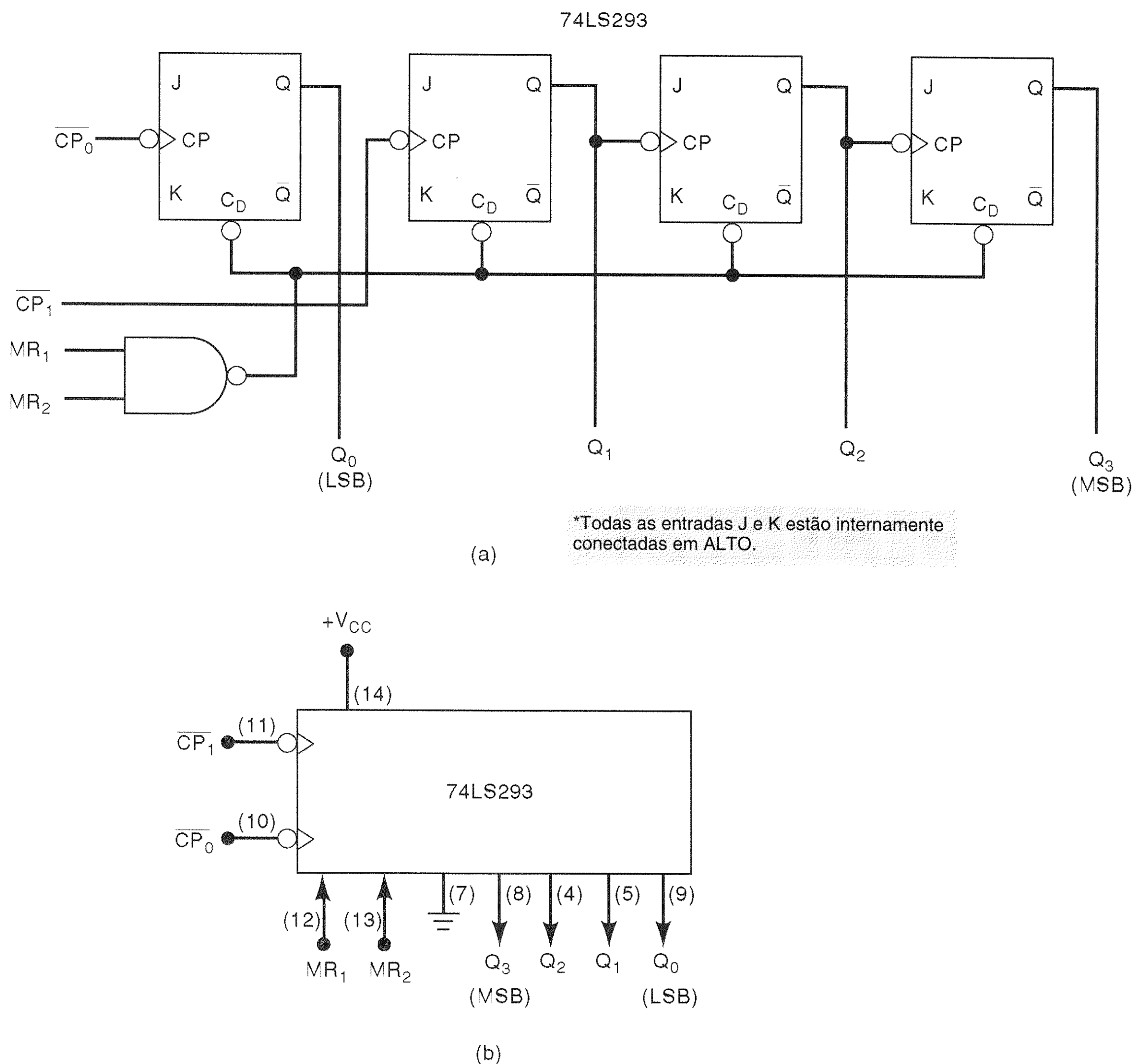


Fig. 7-8 - (a) Diagrama lógico para o CI contador assíncrono 74LS293; (b) símbolo com os números dos pinos entre parênteses. (Cortesia da Fairchild, uma companhia da Schlumberger.)

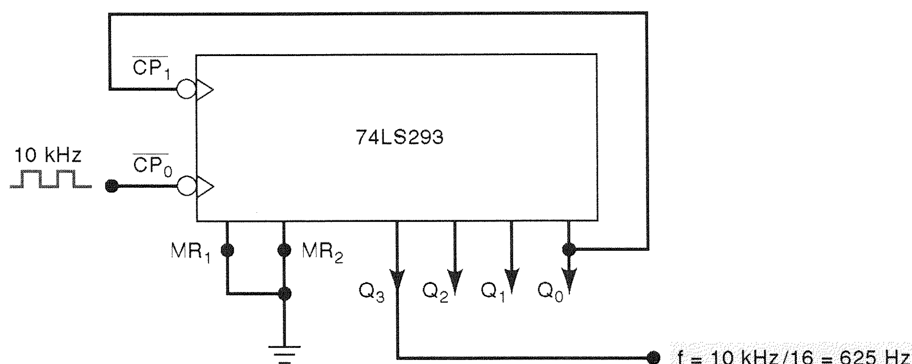


Fig. 7-9 74LS293 ligado como um contador de módulo 16.

Solução
Um contador de módulo 16 requer quatro FFs, e portanto devemos conectar a saída Q_0 em $\overline{CP}_1$, a entrada de clock do flip-flop Q_1 (vide Fig. 7-9). Os pulsos de 10 kHz são aplicados em $\overline{CP}_0$, o clock de entrada de Q_0. A saída é tomada em Q_3.

EXEMPLO 7-9

Mostre como ligar o 74LS293 como um contador de módulo 10.

Solução
Um contador de módulo 10 requer quatro FFs, e portanto mais uma vez precisamos conectar Q_0 em $\overline{CP}_1$. Agora, no entanto, queremos que o contador retorne a 0000 quando ele tentar ir para a contagem 1010 (10). Logo, as saídas Q_3 e Q_1 devem ser conectadas nas entradas de reset; quando ambas vão para ALTO na contagem 1010, a saída da NAND vai imediatamente ressetar o contador para 0000.

As conexões do circuito estão na Fig. 7-10. O diagrama de transição de estados também é apresentado. Note que o estado temporário 1010 não é mostrado.

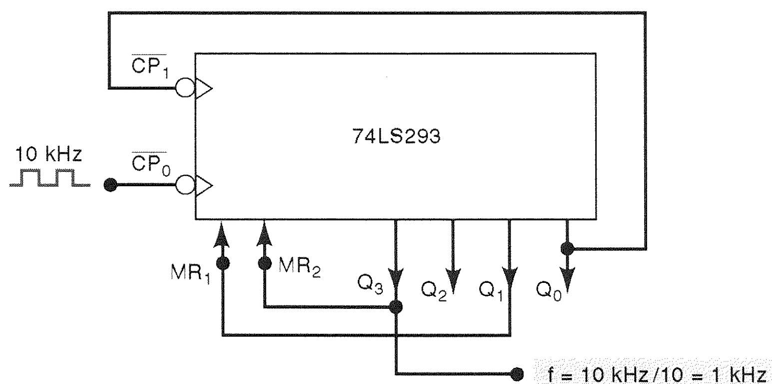
EXEMPLO 7-10

Mostre como ligar um 74LS293 como um contador de módulo 14.

Solução
Quando o contador alcança a contagem de 1110 (14), as saídas Q_3, Q_2 e Q_1 estão todas em ALTO. Infelizmente, o NAND embutido para reset do 74LS293 tem apenas duas entradas. Logo, devemos adicionar uma lógica extra para garantir que o contador seja ressetado de volta para 0000 quando $Q_3 = Q_2 = Q_1 = 1$. De fato, tudo de que precisamos é de uma porta AND de duas entradas, conforme mostrado na Fig. 7-11. Você deve verificar que este arranjo opera como um contador de módulo 14.

EXEMPLO 7-11

No Exemplo 7-7, dividimos a freqüência de entrada por 60 com um contador de módulo 60 usando seis flip-flops J-K e uma porta NAND. Uma outra maneira de se obter um contador de módulo 60 é mostrada na Fig. 7-12. Explique como este circuito funciona.



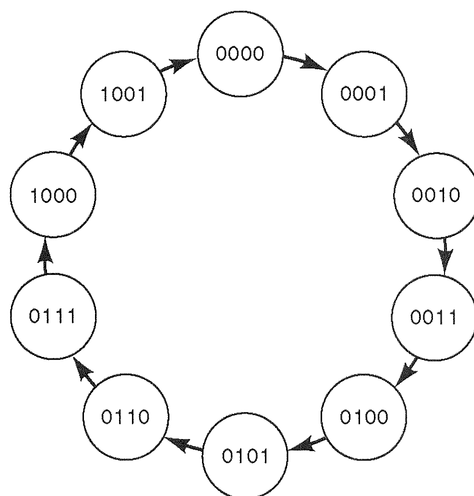


Fig. 7-10 - 74LS293 ligado como um contador de módulo 10.

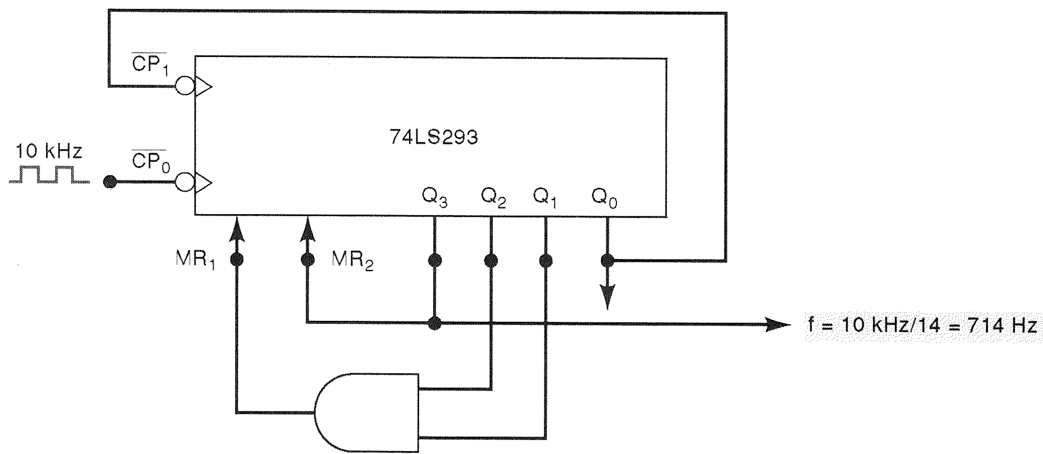


Fig. 7-11 Uma porta AND externa é necessária para ligar o 74LS293 como um contador de módulo 14.

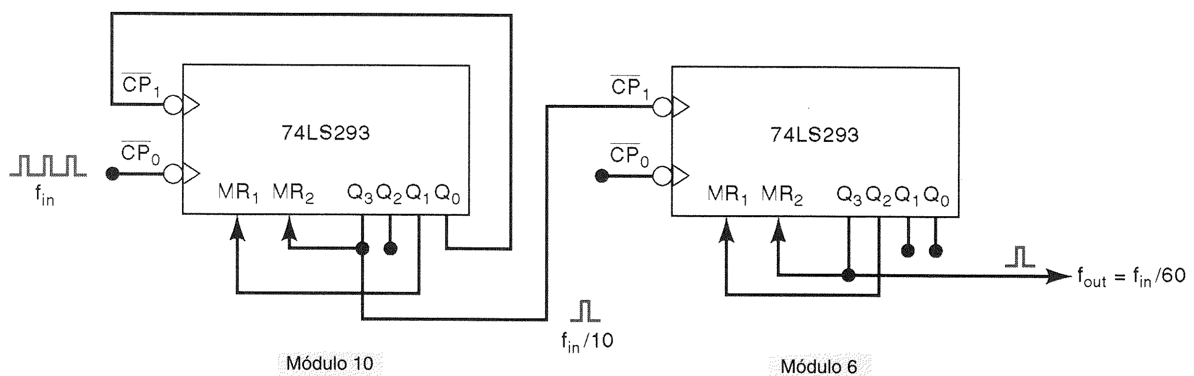


Fig. 7-12 - Dois 74LS293s combinados para fornecer uma divisão de freqüência por 60 através de divisões sucessivas por 10 (módulo 10) e por 6 (módulo 6). Note que para o módulo 6 não foi usado $\overline{CP_0}$ nem Q_0; usavam-se apenas Q_3, Q_2 e Q_1.

Solução

Este circuito divide a freqüência de entrada por 60 em dois passos. O contador 74LS293 da esquerda está ligado como um contador de módulo 10, de modo que sua saída Q_3 tem uma freqüência de $f_{in}/10$. Este sinal é conectado na entrada $\overline{CP_1}$ do segundo contador 74LS293, que está ligado como um contador de módulo 6 (note que Q_0 não está sendo usado). Assim, a saída Q_3 do segundo contador terá a seguinte freqüência:

$$f_{out} = \frac{f_{in}/10}{6} = \frac{f_{in}}{60}$$

O Exemplo 7-11 mostra que dois (ou mais) contadores podem ser ligados em cascata para produzir um módulo total igual ao *produto* de seus módulos individuais. Isto pode ser muito útil em aplicações em que uma divisão de freqüência muito grande seja necessária.

Símbolo IEEE/ANSI para o Contador 74LS293

A Fig. 7-13 mostra o símbolo IEEE/ANSI para o 74LS293. Este símbolo contém vários aspectos novos do padrão IEEE/ANSI. À medida que os descrevermos, você deve observar como a nova simbologia IEEE/ANSI foi concebida para informar-nos muito sobre a operação do CI.

O símbolo contém três blocos distintos. O bloco superior (com as reentrâncias) é o bloco de controle comum. A notação "CTR" define este CI como um contador. Lembre-se

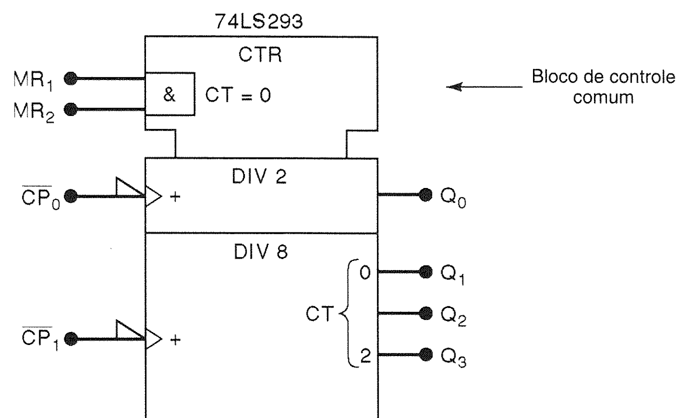


Fig. 7-13 - Símbolo IEEE/ANSI para o CI 74LS293.

da nossa discussão no Cap. 5, de que o bloco de controle comum é usado sempre que um CI tem uma ou mais entradas que são comuns a mais de um circuito no chip. Para o 74LS293, as entradas MR_1 e MR_2 são comuns a todos os FFs do contador.

MR_1 e MR_2 são mostradas como entradas ativas em nível ALTO, que são internamente combinadas usando a operação AND, conforme está indicado pela notação "&". Isto indica que *tanto* MR_1 *quanto* MR_2 devem estar em seus estados ativos para limpar o contador. A notação "CT = 0" nos informa que a ação das entradas *MR* é tornar a contagem igual a zero.

O bloco intermediário está identificado como "DIV2" para indicar que é um contador de módulo 2, que obviamente é um simples FF. DIV2 significa que o contador dividirá sua freqüência de entrada do clock por 2. O bloco inferior está identificado como "DIV8" para indicar que é um contador de módulo 8. As entradas de clock para cada um destes blocos estão mostradas como sendo ativadas nas transições de descida. A notação "+" em cada entrada de clock indica que a descida do clock causará o *incremento de 1* na contagem. Em outras palavras, este é um **contador crescente**, significando que ele *conta para cima* em cada descida. Uma notação "−" seria usada para um **contador decrescente** (vide Seção 7-4).

Embora continuando a usar a simbologia tradicional para o 74LS293 e outros contadores, usaremos alguma notação da simbologia IEEE/ANSI. Por exemplo, indicaremos o módulo de um contador utilizando a notação DIV*n*, onde *n* é o módulo.

Contadores Assíncronos CMOS

Existem diversos contadores assíncronos na família CMOS. A maioria deles é equivalente às versões TTL. Entretanto, alguns CIs de contadores assíncronos CMOS não têm um TTL correspondente. Um destes é o 74HC4024; seu símbolo lógico é mostrado na Fig. 7-14. Ele é um contador de sete bits com uma entrada de reset geral assíncrona. Os sete FFs estão internamente conectados como um contador por pulsação de módulo 128. A entrada *MR* é ativa em ALTO e pode ser usada para ressetar todos os FFs para o estado 0. Note que utilizamos a notação "CTR DIV128" para significar que é um contador de módulo 128.

Fig. 7-14 - Símbolo lógico para o contador por pulsação CMOS 74HC4024. O sinal + indica operação de contagem crescente.

Um outro contador por pulsação CMOS sem correspondente TTL é o 74HC4040, que é um contador de 12 bits com uma entrada de reset geral. A entrada de clock deste contador é do tipo Schmitt-trigger, o que permite a utilização de sinais que variam lentamente sem produzir problemas de contagem.

> **Questões de Revisão**
>
> **1.** Um sinal de clock de 2 kHz é aplicado em $\overline{CP}_1$ de um 74LS293. Qual é a freqüência em Q_3?
> **2.** Qual seria a freqüência final de saída se a ordem dos contadores fosse trocada na Fig. 7-12?
> **3.** Qual é o módulo do contador 74HC4040?
> **4.** O que deve significar a notação "DIV64" em um símbolo de contador?
> **5.** Conecte Q_6 e Q_5 nas entradas de uma porta AND na Fig. 7-14, e conecte a saída da AND em *MR*. Qual será o novo módulo?

7-4 CONTADOR ASSÍNCRONO DECRESCENTE

Todos os contadores que temos estudado até agora contam de modo progressivo *a partir* de zero, isto é, eles são **contadores crescentes**. É relativamente simples construir **contadores decrescentes** assíncronos (por pulsação), que contarão de modo regressivo a partir de um valor máximo até zero. Antes de analisarmos o circuito para um contador decrescente por pulsação, vamos examinar a seqüência de contagem decrescente para um contador de três bits:

	CBA	
(7)	111 ←	
(6)	110	
(5)	101	
(4)	100	Recicla
(3)	011	
(2)	010	
(1)	001	
(0)	000	

A, *B* e *C* representam os estados de saída dos FFs conforme o contador avança na sua seqüência. Pode ser constatado que o flip-flop *A* (LSB) muda de estado (comuta) a cada passo da seqüência do mesmo modo que num contador crescente. O flip-flop *B* troca de estado cada vez que *A* vai de BAIXO para ALTO; *C* muda de estado cada vez que *B* vai de BAIXO para ALTO. Logo, em um contador decrescente, cada FF, exceto o primeiro, deve comutar quando o FF precedente vai de BAIXO para ALTO. Se os FFs têm entradas *CLK* sensíveis na descida (ALTO para BAIXO), então um inversor pode ser colocado para cada entrada *CLK*; entretanto, o mesmo efeito pode ser conseguido acionando cada entrada *CLK* dos FFs com a saída *invertida* do FF precedente. Isto está ilustrado na Fig. 7-15 para um contador decrescente de módulo 8.

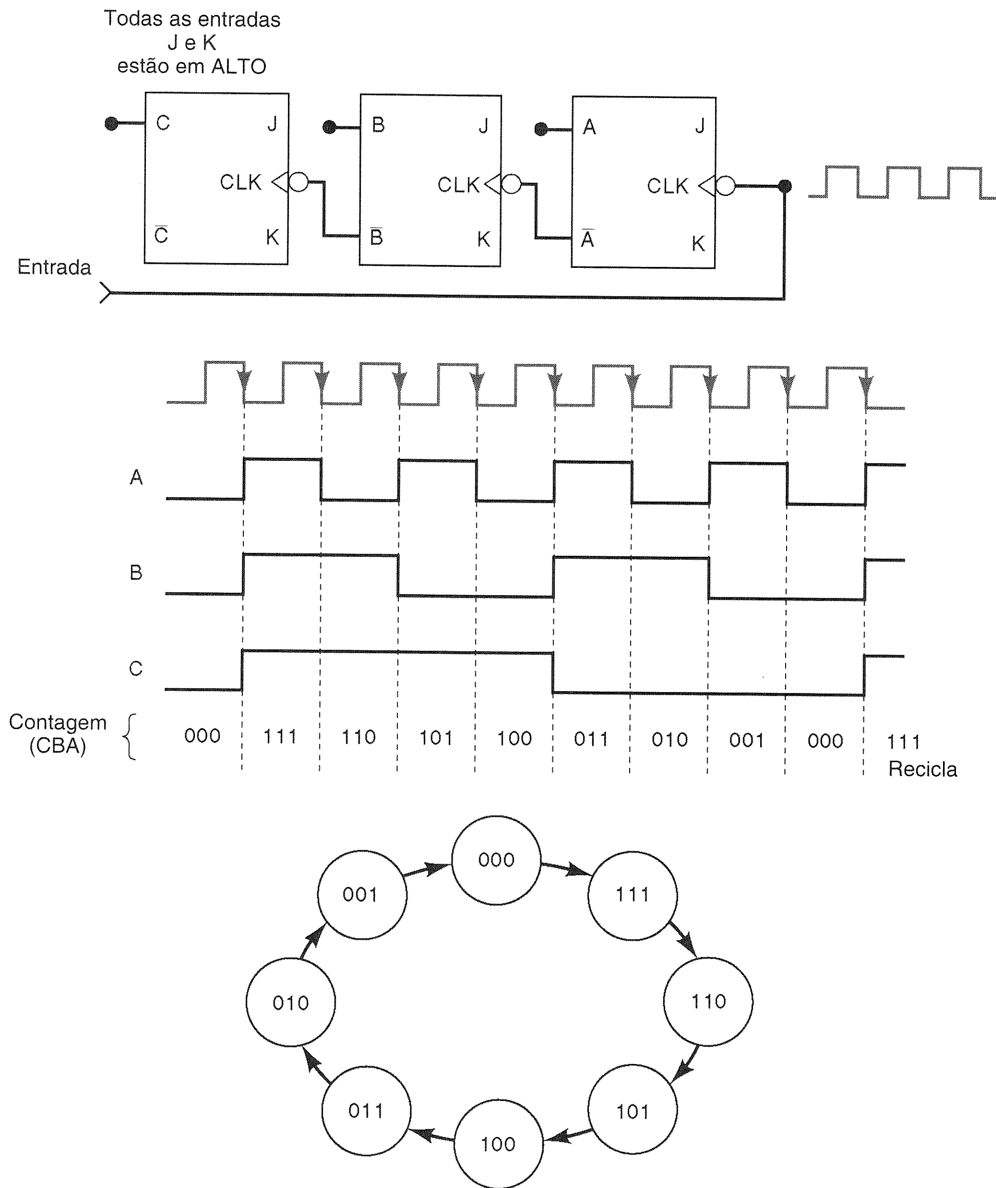


Fig. 7-15 - Contador decrescente de módulo 8.

Os pulsos de entrada são aplicados no flip-flop *A*. A saída $\overline{A}$ aciona a entrada de clock do flip-flop *B*; a saída $\overline{B}$ aciona a entrada de clock do flip-flop *C*. As formas de onda em *A*, *B* e *C* mostram que *B* comuta sempre que *A* vai de BAIXO para ALTO (portanto $\overline{A}$ vai de ALTO para BAIXO) e *C* comuta sempre que *B* vai de BAIXO para ALTO. Isto resulta na desejada seqüência decrescente de contagem nas saídas *C*, *B* e *A*. O diagrama de transição de estados mostra a seqüência. Compare-o com o diagrama para o contador crescente de módulo 8 na Fig. 5-49.

Contadores decrescentes não são tão amplamente usados como os contadores crescentes. Sua principal aplicação é em situações onde se deve reconhecer quando um número desejado de pulsos de entrada ocorreu. Nestas situações, o contador decrescente é *inicializado* com o número desejado e então habilitado a contar para baixo conforme os pulsos são aplicados. Quando o contador alcança o estado *zero*, isto é detectado por uma porta lógica cuja saída indica que o número de pulsos já ocorreu. Discutiremos contadores com carga paralela na Seção 7-8.

Questões de Revisão

1. Qual é a diferença entre as seqüências de contagem de um contador crescente e de um contador decrescente?
2. Descreva como um circuito de contador assíncrono decrescente difere de um circuito de contador crescente.

7-5 ATRASO DE PROPAGAÇÃO EM CONTADORES ASSÍNCRONOS

Contadores assíncronos são o tipo mais simples de contadores binários, já que eles necessitam de um menor número de componentes para produzir uma certa operação de contagem. Eles têm, no entanto, uma grande desvantagem, que é causada pelo seu princípio básico de operação: cada FF é disparado pela transição na saída do FF precedente. Por causa do inerente atraso de propagação (t_{pd}) de cada FF, isto significa que o segundo FF não responderá até um tempo t_{pd} após o primeiro FF ter recebido uma transição ativa do clock; o terceiro FF não responderá até um tempo igual a $2 \times t_{pd}$ após aquela transição do clock; e assim por diante. Em outras palavras, os atrasos de propagação dos FFs se acumulam, de modo que o enésimo FF não pode mudar de estado até um tempo igual a $N \times t_{pd}$ após a transição do clock ter ocorrido. Isto é ilustrado na Fig. 7-16, onde as formas de onda para um contador assíncrono de três bits são mostradas.

O primeiro conjunto de formas de onda na Fig. 7-16(a) mostra uma situação onde um pulso de entrada ocorre a cada 1000 ns (o período do clock é $T = 1000$ ns) e considerou-se que cada FF tem um atraso de propagação de 50 ns ($t_{pd} = 50$ ns). Note que a saída do flip-flop A comuta 50 ns após a descida de cada pulso de entrada. Analogamente, B comuta 50 ns depois de A ter ido de 1 para 0, e C comuta 50 ns após B ter ido de 1 para 0. Como resultado, temos que, quando a quarta transição de descida da entrada ocorre, a saída C vai para ALTO após um atraso de 150 ns. Nesta situação, o contador opera adequadamente no sentido de que os FFs acabam indo para os seus estados corretos, representando a contagem binária. No entanto, a situação piora se os pulsos de entrada são aplicados com uma freqüência bem mais alta.

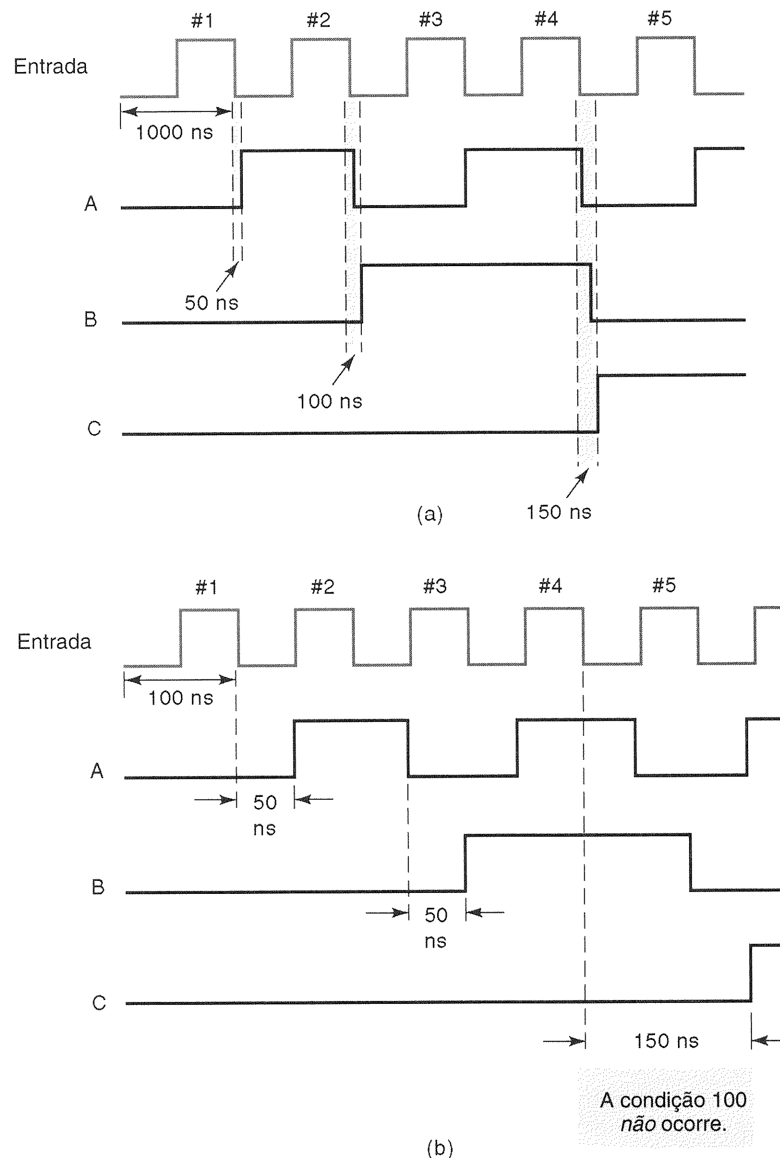


Fig. 7-16 - Formas de onda para um contador assíncrono de três bits ilustrando os efeitos dos atrasos de propagação dos FFs para diferentes freqüências dos pulsos de entrada.

As formas de onda na Fig. 7-16(b) mostram o que acontece se os pulsos de entrada ocorrerem a cada 100 ns. Novamente, cada saída de FF responde 50 ns depois da transição de 1 para 0 na sua entrada *CLK* (note a mudança na escala de tempo). De particular interesse é a situação após a borda de descida do *quarto* pulso de entrada, onde a saída *C* não vai para ALTO até 150 ns mais tarde, que é o mesmo tempo que a saída *A* vai para ALTO em resposta ao *quinto* pulso de entrada. Em outras palavras, a condição $C = 1$, $B = A = 0$ (contagem 100) nunca aparece, pois a freqüência de entrada é muito elevada. Isto poderia causar sérios problemas se esta condição fosse usada para controlar alguma outra operação em um sistema digital. Problemas como este podem ser evitados se o período entre os pulsos é bem maior do que o atraso de propagação total do contador. Isto é, para a operação apropriada do contador precisamos que

$$T_{clock} \geq N \times t_{pd} \qquad (7\text{-}2)$$

onde N = número de FFs. Em termos de freqüência de entrada do clock, a máxima freqüência que pode ser usada é dada por

$$f_{max} = \frac{1}{N \times t_{pd}} \qquad (7\text{-}3)$$

Por exemplo, suponha que um contador binário assíncrono de 4 bits é construído usando o flip-flop J-K 74LS112. A Tabela 5-2 mostra que o 74LS112 tem t_{PLH} = 16 ns e t_{PHL} = 24 ns como sendo os atrasos de propagação do *CLK* para *Q*. Para calcular a f_{max} vamos considerar o "pior caso", isto é, usaremos $t_{pd} = t_{PHL}$ = 24 ns, de modo que

$$f_{max} = \frac{1}{4 \times 24 \text{ ns}} = 10{,}4 \text{ MHz}$$

Certamente, conforme o número de FFs no contador aumenta, o atraso de propagação total aumenta, e f_{max} diminui. Por exemplo, um contador assíncrono que usa seis FFs 74LS112 terá

$$f_{max} = \frac{1}{6 \times 24 \text{ ns}} = 6{,}9 \text{ MHz}$$

Assim, contadores assíncronos não são úteis em freqüências muito altas, especialmente para um grande número de bits. Um outro problema causado pelos atrasos de propagação em contadores assíncronos ocorre quando a saída do contador é *decodificada*. Este problema é discutido na Seção 7-12. Apesar destes problemas, a simplicidade dos contadores assíncronos torna-os muito úteis em aplicações onde sua limitação de freqüência não é crítica.

Questões de Revisão

1. Explique por que o limite máximo de freqüência dos contadores assíncronos diminui conforme mais FFs são adicionados ao contador.
2. Um determinado flip-flop J-K tem t_{pd} = 12 ns. Qual é o maior módulo que pode ser obtido construindo-se um contador a partir destes FFs que opere até 10 MHz?

7-6 CONTADORES SÍNCRONOS

Os problemas encontrados nos contadores assíncronos são causados pela acumulação dos atrasos de propagação dos FFs. Apresentando de outro modo, os FFs não trocam de estado simultaneamente em sincronia com os pulsos de entrada. Estas limitações podem ser superadas com a utilização de **contadores síncronos** ou **contadores paralelos**, nos quais todos os FFs são disparados simultaneamente (em paralelo) pelos pulsos de clock da entrada. Como os pulsos de entrada são aplicados em todos os FFs, algum modo deve ser usado para controlar quando um FF comuta e quando ele deve permanecer inalterado pelo pulso de clock. Isto é conseguido pela utilização das entradas *J* e *K*, e está ilustrado na Fig. 7-17 para um contador síncrono de quatro bits com módulo 16.

Se compararmos o circuito para este contador síncrono com o correspondente assíncrono na Fig. 7-1, podemos constatar as seguintes diferenças marcantes:

- As entradas *CLK* de todos os FFs estão conectadas juntas, de modo que o sinal de entrada do clock é aplicado em cada FF simultaneamente.
- Apenas o flip-flop *A*, o LSB, tem suas entradas *J* e *K* permanentemente em nível ALTO. As entradas *J* e *K* dos outros FFs são acionadas por alguma combinação das saídas dos FFs.
- O contador síncrono necessita de mais circuitos do que um contador assíncrono.

Operação do Circuito

Para este circuito contar apropriadamente numa dada descida do clock, apenas aqueles FFs que comutariam naquela descida deveriam ter $J = K = 1$ quando a transição ocorrer. Vamos analisar a seqüência de contagem na Fig. 7-17(b) para ver o que isso significa para cada FF.

A seqüência de contagem mostra que o flip-flop *A* deve trocar de estado em cada descida. Por isso, suas entradas *J* e *K* estão permanentemente em ALTO, de modo que ele comutará em cada descida do clock de entrada.

A seqüência de contagem mostra que o flip-flop *B* deve mudar de estado em cada descida que ocorrer enquanto $A = 1$. Por exemplo, quando a contagem for 0001, a próxima descida deve comutar *B* para o estado 1; quando a contagem for 0011, a próxima descida deve comutar *B* para o estado 0, e assim por diante. Esta operação é conseguida conectando-se a saída *A* nas entradas *J* e *K* do flip-flop *B*, de modo que $J = K = 1$ somente quando $A = 1$.

A seqüência de contagem mostra que o flip-flop *C* deve mudar de estado em cada descida que ocorrer enquanto $A = B = 1$. Por exemplo, quando a contagem for 0011, a próxima descida deve comutar *C* para o estado 1; quando a contagem for 0111, a próxima descida deve comutar *C* para o estado 0, e assim por diante. Conectando-se o sinal lógico *AB* nas entradas *J* e *K* do flip-flop *C*, este FF somente comutará quando $A = B = 1$.

Analogamente, podemos constatar que o flip-flop *D* deve comutar em toda descida que ocorrer enquanto $A = B = C = 1$. Quando a contagem for 0111, a próxima descida deve comutar *D* para o estado 1; quando a contagem for 1111, a

(a)

Contagem	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
0	0	0	0	0
.	.	.	.	.
.	.	.	.	.
.	.	etc.	.	.

(b)

Fig. 7-17 - Contador síncrono de módulo 16. Cada FF é disparado pela descida do sinal de clock de entrada, de modo que todas as transições dos FFs ocorrem ao mesmo tempo.

próxima descida deve comutar D para o estado 0. Conectando-se o sinal lógico ABC nas entradas J e K do flip-flop D, este flip-flop comutará somente quando $A = B = C = 1$.

O princípio básico para a construção de um contador síncrono pode ser enunciado como segue:

Cada FF deve ter suas entradas *J* e *K* conectadas de modo que elas estejam em ALTO somente quando as saídas de todos os FFs de mais baixa ordem estiverem no estado ALTO.

Vantagem dos Contadores Síncronos sobre os Assíncronos

Em um contador paralelo todos os FFs mudarão de estado simultaneamente, isto é, todos eles estão sincronizados com a descida dos pulsos de clock de entrada. Logo, ao contrário dos contadores assíncronos, os atrasos de propagação dos FFs não se acumulam para produzir um atraso geral. Em vez disso, o tempo de resposta total de um contador síncrono, como o da Fig. 7-17, é o tempo que leva *um* FF para comutar, mais o tempo para o novo nível lógico se propagar através de *uma* única porta AND, para alcançar as entradas J e K. Isto é, para um contador síncrono,

$$\textbf{atraso total} = t_{pd} \textbf{ do FF} + t_{pd} \textbf{ da porta AND}$$

Este atraso total é o mesmo, não importando quantos FFs estão no contador, e ele será geralmente bem menor do que o obtido em um contador assíncrono com o mesmo número de FFs. Logo, um contador síncrono pode operar a uma freqüência de entrada bem maior. Naturalmente, o circuito para o contador síncrono é mais complexo do que aquele para o contador assíncrono.

CIs Reais

Existem diversos CIs de contadores síncronos tanto na família lógica TTL quanto na CMOS. Alguns dos dispositivos mais comumente utilizados são:

- 74LS160/162, 74HC160/162: contadores síncronos decádicos
- 74LS161/163, 74HC161/163: contadores síncronos de módulo 16

EXEMPLO 7-12

(a) Determine f_{max} para o contador da Fig. 7-17(a) se o t_{pd} para cada FF é 50 ns e o t_{pd} para cada porta AND é 20 ns. Compare este valor com f_{max} para um contador assíncrono de módulo 16.

(b) O que deve ser feito para converter este contador em um que tenha módulo 32?

(c) Determine a f_{max} para o contador paralelo de módulo 32.

Solução

(a) O atraso total que deve ser tolerado entre os pulsos de entrada do clock é igual a t_{pd} do FF + t_{pd} da porta AND. Logo, $T_{clock} \geq 50 + 20 = 70$ ns, * e portanto o contador paralelo tem

$$f_{max} = \frac{1}{70 \text{ ns}} = 14{,}3 \text{ MHz (contador síncrono)}$$

Um contador assíncrono de módulo 16 utiliza quatro FFs com $t_{pd} = 50$ ns. Logo, f_{max} para o contador assíncrono é

$$f_{max} = \frac{1}{4 \times 50 \text{ ns}} = 5 \text{ MHz (contador assíncrono)}$$

(b) Um quinto FF deve ser incluído, visto que $2^5 = 32$. A entrada *CLK* deste FF também é ligada aos pulsos de entrada. Suas entradas *J* e *K* são acionadas pela saída de uma porta AND cujas quatro entradas são *A*, *B*, *C* e *D*.

(c) f_{max} ainda é determinada como no item (a) independentemente do número de FFs no contador paralelo. Logo, f_{max} ainda é 14,3 MHz.

Questões de Revisão

1. Qual é a vantagem de um contador síncrono sobre um contador assíncrono? Qual é a desvantagem?
2. Quantos dispositivos lógicos são necessários para um contador paralelo de módulo 64?
3. Que sinal lógico aciona as entradas *J* e *K* do flip-flop MSB do contador da questão 2?

*O autor desprezou o tempo de setup das entradas *J* e *K*. (*N. T.*)

7-7 CONTADORES SÍNCRONOS DECRESCENTES E CRESCENTES/DECRESCENTES

Na Seção 7-4, vimos que um contador assíncrono poderia contar de modo decrescente utilizando-se a saída invertida de cada FF para acionar o próximo FF do contador. Um contador paralelo decrescente pode ser construído de modo similar, isto é, utilizando-se as saídas invertidas de cada FF para acionar as entradas *J* e *K* seguintes. Por exemplo, o contador crescente paralelo da Fig. 7-17 pode ser convertido para decrescente conectando-se as saídas $\overline{A}$, $\overline{B}$ e $\overline{C}$ em vez de *A*, *B* e *C*, respectivamente. O contador então contará 15, 14, 13, 12, ..., 3, 2, 1, 0, 15, 14, 13, e assim por diante.

A Fig. 7-18(a) mostra como fazer um **contador crescente/decrescente** (up/down). A entrada Up/$\overline{\text{Down}}$ controla se as saídas normais ou as invertidas dos FFs são conectadas nas entradas *J* e *K* dos sucessivos FFs. Quando Up/$\overline{\text{Down}}$ é mantida em ALTO, as portas AND 1 e 2 são habilitadas, enquanto as portas 3 e 4 estão desabilitadas (note o inversor). Isto permite que as saídas *A* e *B*, através das portas 1 e 2, alcancem as entradas *J* e *K* dos FFs *B* e *C*. Quando Up/$\overline{\text{Down}}$ é mantido em nível BAIXO, as portas AND 1 e 2 são desabilitadas, enquanto as portas AND 3 e 4 são habilitadas. Isto permite que as saídas *A* e *B*, através das portas 3 e 4, alcancem as entradas *J* e *K* dos FFs *B* e *C*. As formas de onda na Fig. 7-18(b) ilustram a operação. Note que para os primeiros cinco pulsos de clock, Up/$\overline{\text{Down}}$ = 1, e a contagem é crescente; para os últimos cinco pulsos, Up/$\overline{\text{Down}}$ = 0, e a contagem é decrescente.

A nomenclatura usada para o sinal de controle (Up/$\overline{\text{Down}}$) foi escolhida para tornar claro como ele afeta o contador. A operação crescente é ativa em ALTO, e a decrescente é ativa em nível BAIXO.

EXEMPLO 7-13

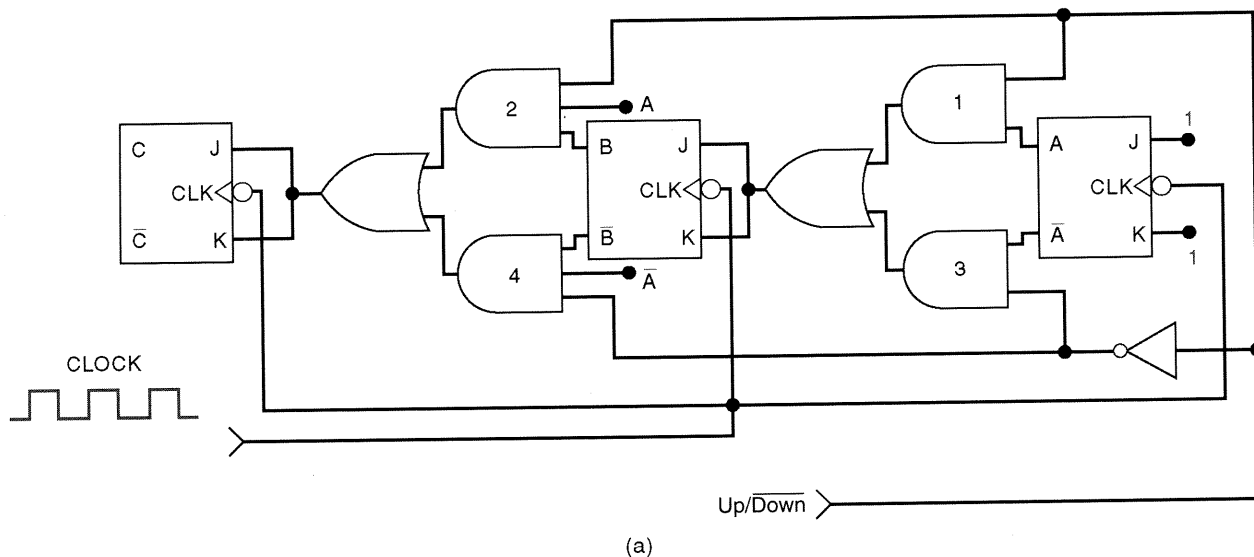
Que problemas poderiam ser causados se o sinal Up/$\overline{\text{Down}}$ mudasse de nível na transição negativa do clock?

Solução

Os FFs poderiam operar de modo imprevisível, visto que alguns deles teriam suas entradas *J* e *K* mudando aproximadamente no mesmo instante de tempo que a transição negativa nas suas entradas *CLK* ocorresse. Entretanto, os efeitos da mudança do sinal de controle devem se propagar através de duas portas antes de alcançar as entradas *J* e *K*, e portanto é mais provável que os FFs respondam de modo previsível aos níveis que estavam anteriormente em *J* e *K* antes da descida de *CLK*.

7-8 CONTADORES COM CARGA PARALELA

Muitos contadores síncronos (paralelos) que estão disponíveis como CIs são projetados para serem **carregáveis**; em outras palavras, eles podem ser inicializados com qualquer contagem inicial desejada, assincronamente (independen-



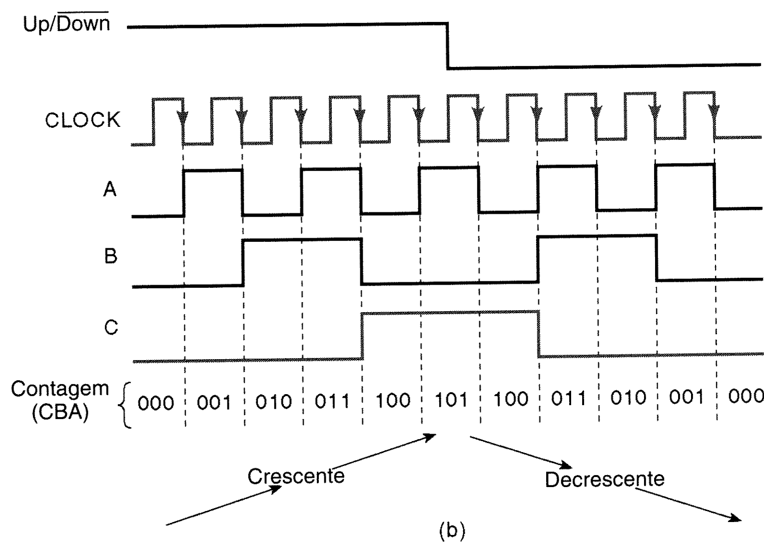


Fig. 7-18 - (a) Contador síncrono crescente/decrescente de módulo 8. (b) O contador conta de modo crescente quando a entrada de controle Up/$\overline{\text{Down}}$ = 1; ele conta decrescente quando a entrada de controle = 0.

temente do sinal de clock) ou de modo síncrono (na transição ativa do sinal de clock). Esta operação de inicialização também é denominada **carga paralela** do contador.

A Fig. 7-19 mostra o circuito lógico para um contador crescente de três bits com carga paralela. As entradas *J*, *K* e *CLK* são ligadas para a operação como contador síncrono crescente. As entradas assíncronas de PRESET e CLEAR estão ligadas para realizar a carga assíncrona. O contador é carregado com qualquer contagem desejada, a qualquer instante, fazendo-se o seguinte:

1. Aplique a contagem desejada nas entradas paralelas de dados, P_2, P_1 e P_0.
2. Aplique um pulso em BAIXO na entrada de CARGA PARALELA (PARALLEL LOAD), $\overline{PL}$.

Este procedimento realizará uma transferência assíncrona dos níveis de P_2, P_1 e P_0 para os flip-flops Q_2, Q_1 e Q_0, respectivamente (Seção 5-17). Esta *transferência forçada* ocorre independentemente das entradas *J*, *K* e *CLK*. O efeito da entrada *CLK* será desabilitado enquanto $\overline{PL}$ ficar no seu estado ativo em BAIXO, visto que, cada FF terá apenas uma de suas entradas assíncronas ativada enquanto $\overline{PL}$ = 0. Uma vez que $\overline{PL}$ retorne para ALTO, os FFs podem responder a suas entradas *CLK* e podem prosseguir a operação de contagem crescente começando do valor que foi carregado no contador.

Por exemplo, digamos que $P_2 = 1$, $P_1 = 0$ e $P_0 = 1$. Enquanto $\overline{PL}$ está em ALTO, estas entradas paralelas de dados não têm efeito algum. Se pulsos de clock estão presentes, o contador realizará a operação normal de contagem crescente. Agora, digamos que $\overline{PL}$ é pulsado em BAIXO quando o contador está com 010 (isto é, $Q_2 = 0$, $Q_1 = 1$ e $Q_0 = 0$). Este nível BAIXO em $\overline{PL}$ produzirá níveis em BAIXO na entrada *CLR* de Q_1 e nas entradas *PRE* de Q_2 e Q_0, e portanto o contador irá para a contagem 101, *independentemente do que esteja ocorrendo na entrada CLK*. A

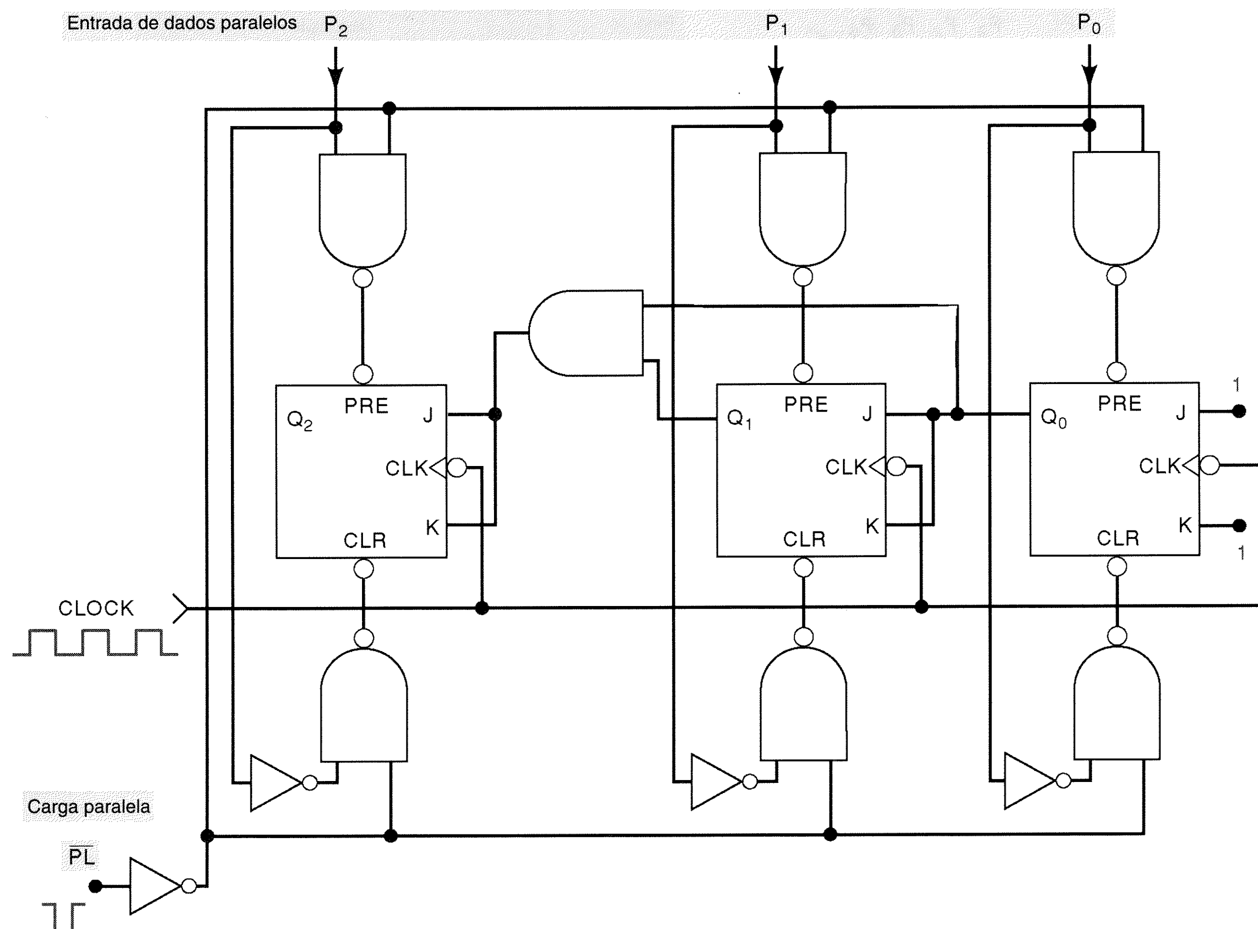


Fig. 7-19 - Contador síncrono com carga paralela assíncrona.

contagem permanecerá em 101 até que $\overline{PL}$ seja desativado (retorne a ALTO); neste instante, o contador prosseguirá a contar os pulsos de modo crescente a partir de 101.

Esta carga assíncrona é usada por muitos CIs de contadores, tais como os TTLs 74LS190, 74LS191, 74LS192 e 74LS193 e os equivalentes CMOS, 74HC190, 74HC191, 74HC192 e 74HC193.

Carga Síncrona

Muitos CIs de contadores paralelos utilizam *carga síncrona,* na qual o contador é carregado na transição ativa do mesmo sinal de clock que é usado para a contagem. O nível lógico aplicado na entrada $\overline{PL}$ determina se a transição ativa do clock vai carregar o contador ou se será contada como na operação normal.

Exemplos de CIs contadores que usam carga síncrona incluem os TTLs 74LS160, 74LS161, 74LS162 e 74LS163 e seus equivalentes CMOS 74HC160, 74HC161, 74HC162 e 74HC163.

Questões de Revisão

1. O que significa dizer que um contador é carregável?

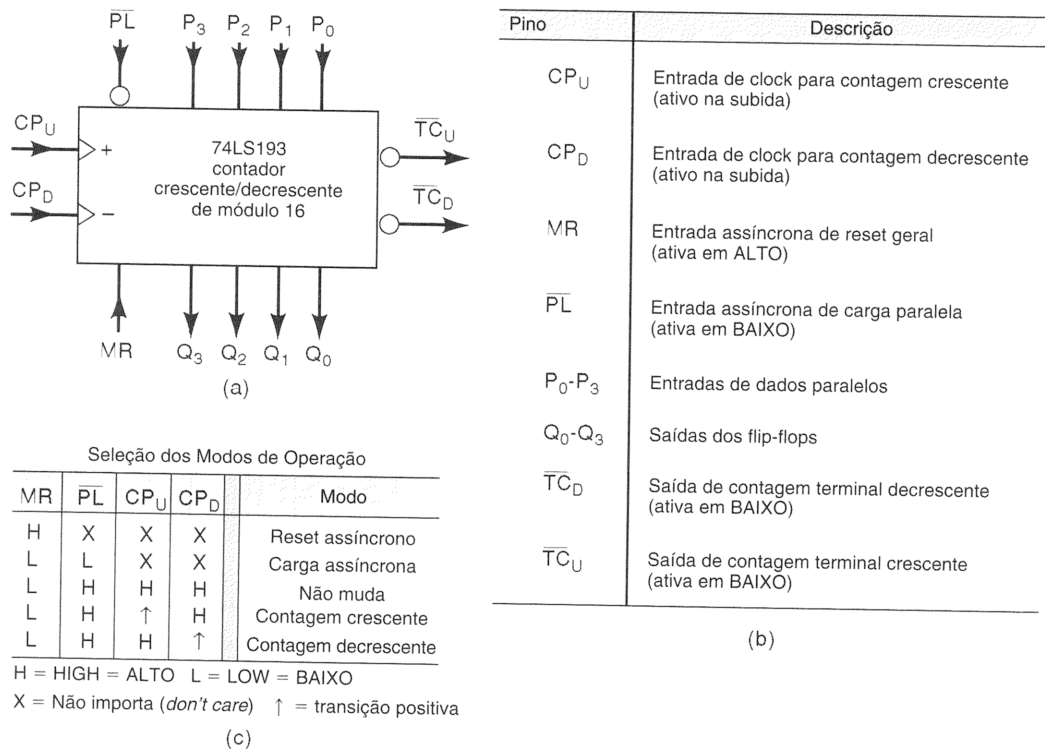
2. Descreva a diferença entre a carga assíncrona e a carga síncrona.

7-9 O 74LS193/HC193

A Fig. 7-20 mostra o símbolo lógico e a descrição das entradas e saídas do contador 74LS193. Este contador pode ser descrito como um contador crescente/decrescente (up/down) síncrono de módulo 16 com carga paralela e reset assíncronos. Vamos agora descrever a função de cada entrada e saída.

Entradas de Clock CP_U e CP_D

O contador vai responder às transições positivas em uma das entradas de clock. CP_U é a entrada de clock para *contagem crescente.* Quando os pulsos são aplicados a esta entrada, o contador vai ser incrementado em cada transição positiva até a contagem máxima 1111, e depois retorna a 0000 para iniciar a contagem novamente. CP_D é a entrada de *clock para contagem decrescente.* Quando os pulsos são aplicados a esta entrada, o contador vai ser decrementado



Pino	Descrição
CP_U	Entrada de clock para contagem crescente (ativo na subida)
CP_D	Entrada de clock para contagem decrescente (ativo na subida)
MR	Entrada assíncrona de reset geral (ativa em ALTO)
$\overline{PL}$	Entrada assíncrona de carga paralela (ativa em BAIXO)
P_0-P_3	Entradas de dados paralelos
Q_0-Q_3	Saídas dos flip-flops
$\overline{TC}_D$	Saída de contagem terminal decrescente (ativa em BAIXO)
$\overline{TC}_U$	Saída de contagem terminal crescente (ativa em BAIXO)

(b)

Seleção dos Modos de Operação

MR	$\overline{PL}$	CP_U	CP_D	Modo
H	X	X	X	Reset assíncrono
L	L	X	X	Carga assíncrona
L	H	H	H	Não muda
L	H	↑	H	Contagem crescente
L	H	H	↑	Contagem decrescente

H = HIGH = ALTO L = LOW = BAIXO
X = Não importa (*don't care*) ↑ = transição positiva

(c)

Fig. 7-20 - Contador crescente/decrescente com carga paralela 74LS193: (a) símbolo lógico; (b) descrição das entradas e saídas; (c) tabela de seleção dos modos de operação. (Cortesia da Fairchild, uma companhia do grupo Schlumberger.)

em cada transição positiva até a contagem mínima 0000, e depois retorna a 1111 para iniciar a contagem novamente. Portanto, apenas uma entrada de clock será usada para contagem, enquanto a outra deverá permanecer inativa (mantida em ALTO).

Reset Geral (*MR*)

O reset geral (master reset) é uma entrada assíncrona e ativa em ALTO que faz com que o contador vá para o estado 0000. *MR* é um reset por nível, e portanto ele fará com que o contador permaneça em 0000 enquanto *MR* for igual a 1. Ele também tem prioridade sobre *todas* as outras entradas.

Entradas de Carga Paralela

Pode-se fazer com que os flip-flops do contador armazenem os níveis lógicos presentes nas entradas de dados paralelas P_3 a P_0, pulsando momentaneamente a entrada de carga paralela $\overline{PL}$ de ALTO para BAIXO. Esta carga é assíncrona e tem prioridade sobre a operação de contagem. Entretanto, $\overline{PL}$ não terá efeito sobre o contador se a entrada *MR* estiver em seu estado ativo em ALTO.

Saídas do Contador

O valor atual da contagem está sempre presente nas saídas dos flip-flops Q_3 a Q_0, onde Q_0 é o LSB e Q_3 é o MSB.

Saídas de Contagem Terminal

As saídas de contagem terminal são utilizadas quando dois ou mais 74LS193 são conectados como um contador de vários estágios para se obter um módulo maior. No modo de contagem crescente, a saída $\overline{TC}_U$ do contador de mais baixa ordem é conectada na entrada CP_U do próximo contador de ordem mais alta. No modo de contagem decrescente, a saída $\overline{TC}_D$ do contador de mais baixa ordem é conectada à entrada CP_D do próximo contador de ordem mais alta.

$\overline{TC}_U$ é a saída de *contagem terminal crescente* (também chamado de *carry*). Ela é gerada no chip 74LS193 utilizando a lógica mostrada na Fig. 7-21(a). Obviamente, $\overline{TC}_U$ estará em BAIXO apenas quando o contador estiver no estado 1111 e CP_U estiver em BAIXO. Assim, $\overline{TC}_U$ permanecerá em ALTO enquanto o contador estiver contando de 0000 a 1110. Na próxima transição positiva de CP_U, o contador vai para o estado 1111, mas $\overline{TC}_U$ não vai para BAIXO até que CP_U retorne a BAIXO. A próxima transição de CP_U faz com que o contador retorne a 0000 e $\overline{TC}_U$ vá para ALTO. Esta transição positiva em $\overline{TC}_U$ ocorre quando o contador vai de 1111 para 0000 e, portanto, pode ser usada para incrementar um segundo 74LS193.

$\overline{TC}_D$ é a saída de *contagem terminal decrescente* (também chamada de *borrow*). Ela é gerada como está mostrado na Fig. 7-21(b). Ela está normalmente em ALTO e não vai para BAIXO até que o contador esteja em 0000 e CP_D esteja em BAIXO. A próxima transição positiva em CP_D faz com que o contador vá para o estado 1111 e $\overline{TC}_D$ retorne a

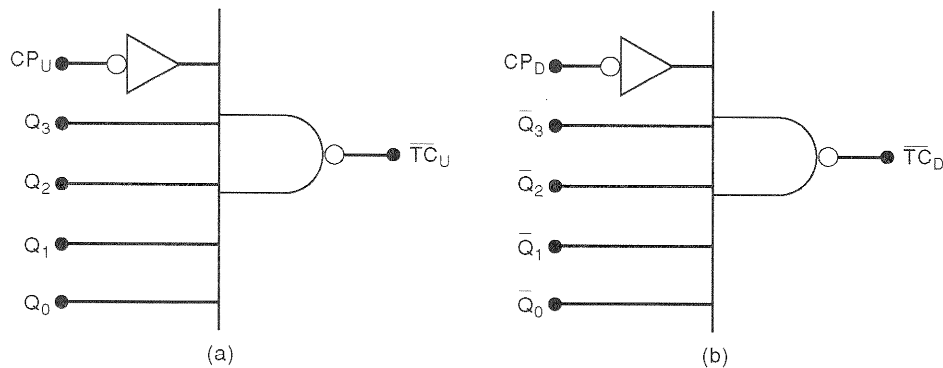


Fig. 7-21 - (a) Lógica no 74LS193 para geração do sinal $\overline{TC}_U$; (b) lógica para geração do sinal $\overline{TC}_D$.

ALTO. Esta transição positiva em $\overline{TC}_D$ pode ser usada para decrementar um segundo 74LS193.

EXEMPLO 7-14

Veja a Fig. 7-22(a), onde um 74LS193 está configurado como um *contador crescente*. As entradas de dados paralelos estão permanentemente conectadas a 1011, e as formas de onda das entradas CP_U, $\overline{PL}$ e MR podem ser vistas na Fig. 7-22(b). Considere que o contador está inicialmente em 0000, e determine as formas de onda das saídas do contador.

Solução

Inicialmente (em t_0) os flip-flops do contador estão todos em BAIXO. Isto faz com que $\overline{TC}_U$ esteja em ALTO. Imediatamente antes do instante t_1, a entrada $\overline{PL}$ é pulsada em

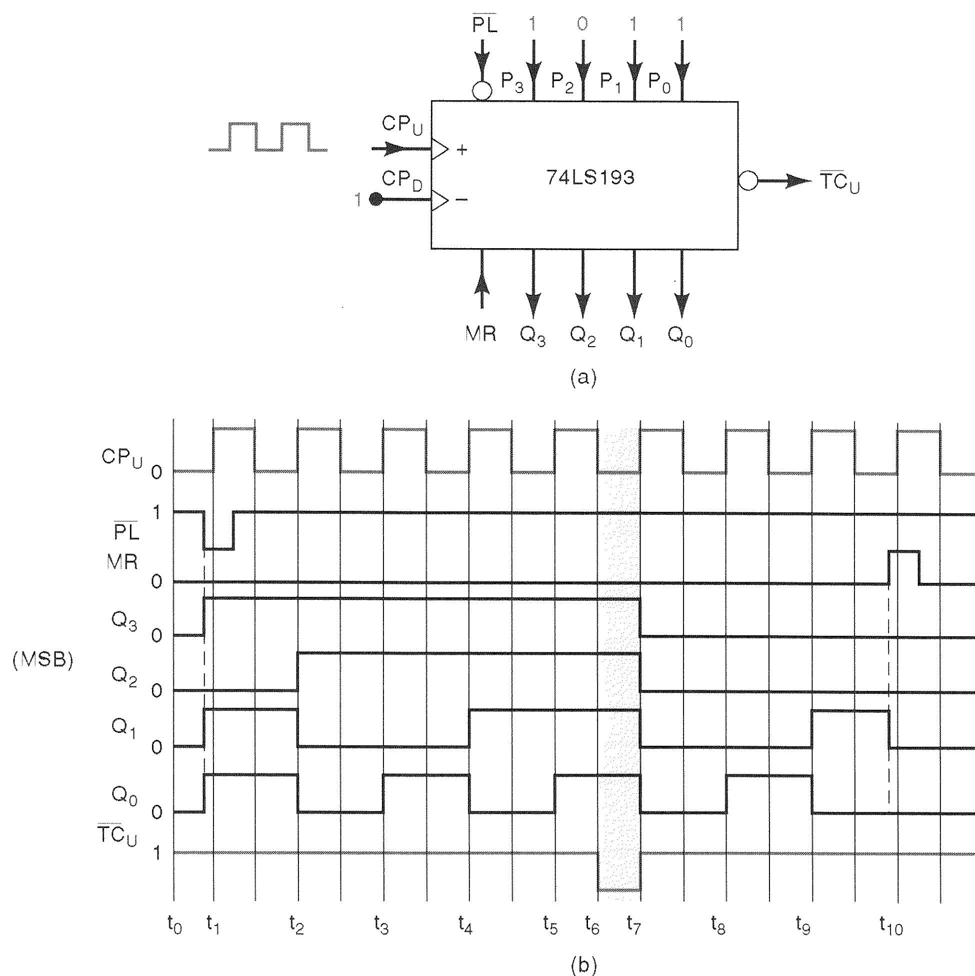


Fig. 7-22 - Exemplo 7-14.

BAIXO. Isso faz com que o contador seja imediatamente carregado com 1011, produzindo $Q_3 = 1$, $Q_2 = 0$, $Q_1 = 1$ e $Q_0 = 1$. No instante t_1, a entrada CP_U faz uma transição positiva, mas o contador não responde a esta transição pois $\overline{PL}$ ainda está ativo. Em t_2, t_3, t_4 e t_5 o contador é incrementado a cada transição positiva de CP_U. Após a transição positiva em t_5 o contador está em 1111, mas $\overline{TC}_U$ não vai para BAIXO até que CP_U vá para BAIXO em t_6. Quando a próxima transição positiva ocorrer em t_7, o contador irá para 0000, e $\overline{TC}_U$ retornará a ALTO.

O contador será incrementado em resposta às transições positivas em t_8 e t_9. A transição positiva em t_{10} não terá efeito algum, porque *MR* vai para ALTO antes do instante t_{10} e permanece ativo em t_{10}. Isto coloca todos os flip-flops em 0 e se sobrepõe ao sinal CP_U.

EXEMPLO 7-15

A Fig. 7-23(a) mostra um 74LS193 configurado como um *contador decrescente*. As entradas paralelas de dados estão permanentemente conectadas em 0111, e as formas de onda de CP_D e $\overline{PL}$ estão mostradas na Fig. 7-23(b). Considere que o contador está inicialmente em 0000 e determine as formas de ondas das saídas.

Solução

Em t_0 todas as saídas estão em BAIXO e CP_D está em BAIXO. Estas condições produzem $\overline{TC}_D = 0$. Antes de t_1, a entrada $\overline{PL}$ é pulsada para BAIXO. Isto imediatamente coloca o contador em 0111 e, portanto, faz com que $\overline{TC}_D$ vá para ALTO. A transição positiva de CP_D em t_1 não terá efeito sobre o contador, uma vez que $\overline{PL}$ ainda está ativo. O contador responderá às transições positivas de t_2 até t_8 e será decrementado para 0000 em t_8. $\overline{TC}_D$ não vai para BAIXO até t_9 quando CP_D vai para BAIXO. Em t_{10} a transição positiva de CP_D faz com que o contador vá para 1111 e também com que $\overline{TC}_D$ retorne a ALTO.

Contador de Módulo Variável Utilizando o 74LS193

Contadores que permitem carga paralela podem ser conectados para se obterem módulos diferentes sem a necessidade de usar circuitos lógicos adicionais. Demonstra-

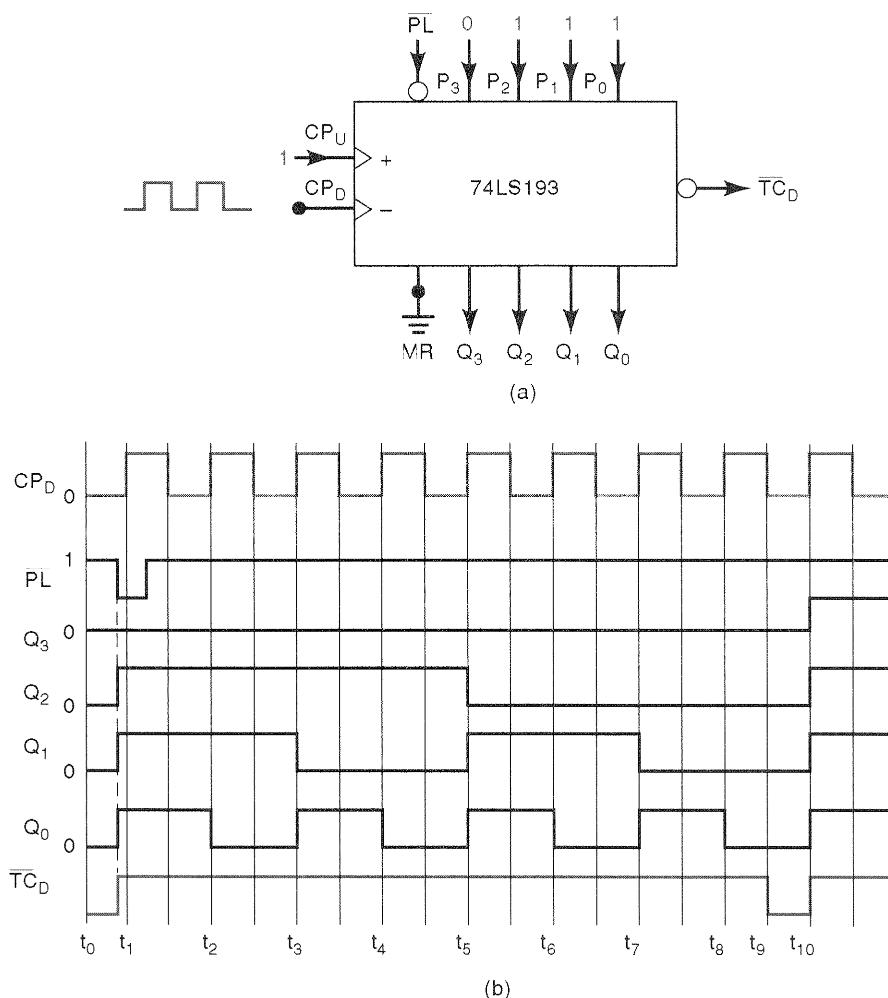


Fig. 7-23 - Exemplo 7-15.

remos esta afirmação para o 74LS193 usando o circuito da Fig. 7-24(a). Neste caso, o 74LS193 é usado como um contador decrescente com suas entradas de carga paralela permanentemente conectadas a 0101 (5_{10}). Observe que a saída $\overline{TC}_D$ está conectada à entrada $\overline{PL}$.

Iniciaremos nossa análise presumindo que o contador está no estado 0101 no instante t_0. Veja na Fig. 7-24(b) as formas de onda dos sinais do contador.

O contador será decrementado nas transições positivas de CP_D nos instantes de t_1 até t_5. Em t_5 o contador está no estado 0000. Quando CP_D vai para BAIXO em t_6, ele força $\overline{TC}_D$ a ir para BAIXO. Isto imediatamente ativa a entrada $\overline{PL}$ e coloca o contador de volta ao estado 0101. Observe que $\overline{TC}_D$ permanece em BAIXO por um pequeno intervalo, pois uma vez que as saídas do contador vão para 0101 em resposta a $\overline{PL} = 0$, a condição necessária para manter $\overline{TC}_D = 0$ é removida. Portanto, existe apenas um pulso estreito em $\overline{TC}_D$.

Esta mesma seqüência é repetida nos instantes t_7 até t_{12} e em intervalos iguais daí em diante. Se examinarmos a forma de onda de Q_2, veremos que ela passa por um ciclo completo a cada *cinco* ciclos de CP_D. Por exemplo, existem *cinco* ciclos de clock entre a transição positiva de Q_2 em t_6 e a transição positiva de Q_2 em t_{11}. Logo, a freqüência da forma de onda de Q_2 é um quinto da freqüência do clock.

Este arranjo possui uma peculiaridade que você pode ter notado: ele conta *seis* diferentes estados (5, 4, 3, 2, 1, 0) e, apesar disto, divide a freqüência por *cinco*. Isto é devido à forma incomum pela qual o contador retorna ao estado 5 no meio do ciclo de clock. Logo, a operação deste contador viola nossa regra geral de que o número de estados e a razão de divisão de freqüência são iguais. Uma vez que este tipo de arranjo é usado principalmente para divisão de freqüência, ignoraremos a seqüência de contagem e diremos que este contador possui módulo igual a 5, uma vez que ele divide a freqüência do clock por 5.

Não é coincidência que a razão de divisão de freqüência é igual ao número aplicado às entradas paralelas de dados (0101 = 5). De fato, podemos variar a divisão de freqüência alterando os níveis lógicos aplicados à entrada paralela de dados.

Um circuito *divisor de freqüência variável* pode ser facilmente implementado conectando-se chaves às entradas paralelas de dados do circuito da Fig. 7-24. As chaves podem ser colocadas em um valor igual ao número pelo qual

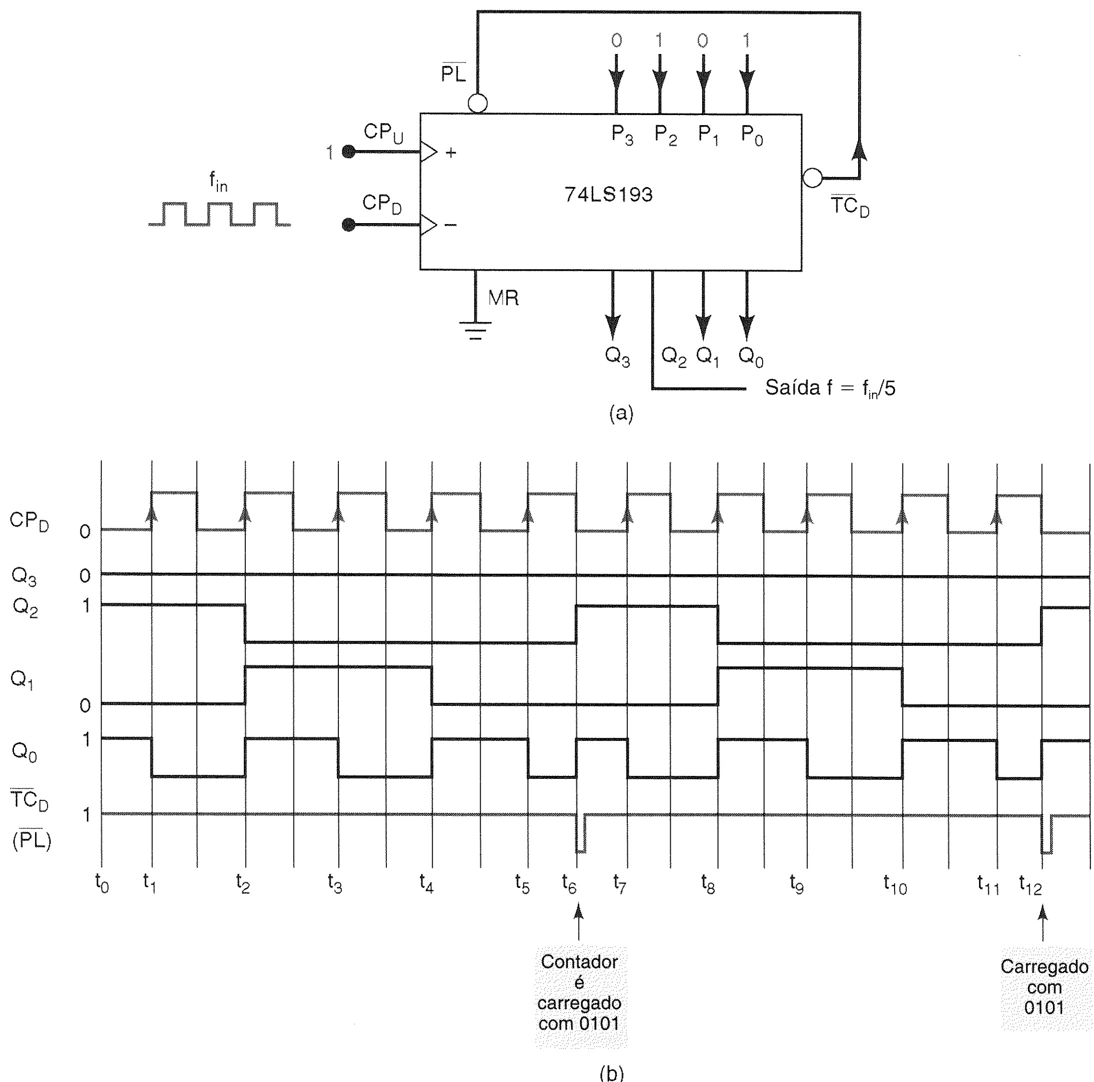


Fig. 7-24 - (a) 74LS193 configurado como um contador de módulo 5; (b) formas de onda.

desejamos dividir a freqüência. Observe que se deve tomar cuidado para escolher a saída Q apropriada, dependendo do valor pelo qual desejamos dividir a freqüência.

Contador com Vários Estágios

Como afirmamos anteriormente, as saídas $\overline{TC}_D$ e $\overline{TC}_U$ são usadas quando dois ou mais 74LS193 são conectados como um **contador de vários estágios**. Na Fig. 7-25, dois contadores estão conectados com um contador crescente/decrescente de dois estágios, o que efetivamente aumenta o intervalo máximo de contagem crescente para 0 → 255 e o intervalo de contagem decrescente para 255 → 0. O bloco à esquerda é o estágio de baixa ordem e é disparado por uma ou outra das entradas de clock. As saídas $\overline{TC}_U$ e $\overline{TC}_D$ deste estágio estão conectadas às entradas de clock do estágio de alta ordem. Observe o uso de uma entrada comum $\overline{\text{Load}}$ e de uma entrada comum Reset. Observe também que as entradas paralelas de dados do estágio de alta ordem são denominadas $P_4P_5P_6P_7$, e as saídas desse estágio estão denominadas $Q_4Q_5Q_6Q_7$. Um número de 8 bits pode ser colocado em um contador de 8 bits, e podemos incrementá-lo ou decrementá-lo a partir da contagem inicial. O valor da contagem em qualquer instante aparece nas saídas Q_0–Q_7.

Questões de Revisão

1. Descreva a função das entradas $\overline{PL}$ e P_0 a P_3.
2. Descreva a função da entrada *MR*.
3. *Verdadeiro ou falso*: O 74LS193 não pode ser carregado enquanto *MR* está ativo.
4. Que níveis lógicos devem estar presentes em CP_D, $\overline{PL}$ e *MR* para que o 74LS193 conte pulsos que apareçam na entrada CP_U?
5. Qual seria o intervalo máximo de contagem para um contador de quatro estágios feito a partir de CIs 74LS193?

7-10 MAIS SOBRE A NOTAÇÃO DE DEPENDÊNCIA IEEE/ANSI*

Podemos aprender mais sobre a notação de dependência, que é uma parte tão importante da nova simbologia IEEE/ANSI, através do exame do símbolo utilizado para o CI 74LS193, que pode ser visto na Fig. 7-26. Cada tipo de CI que examinarmos que utilizar esta nova simbologia vai ajudá-lo a melhor compreendê-la e prepará-lo para uma utilização mais intensa destes símbolos no futuro.

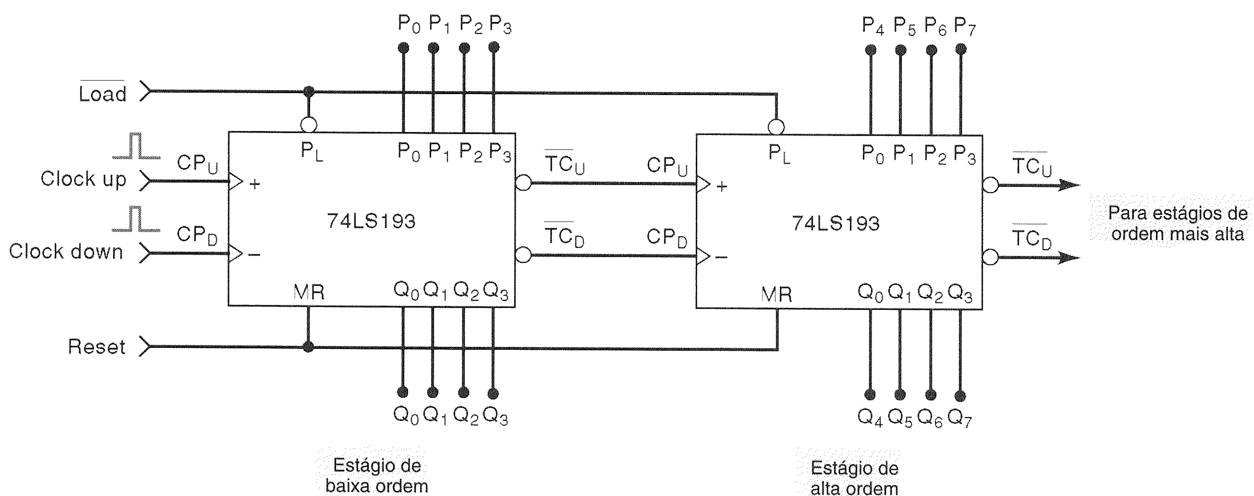
Mais uma vez, devemos lembrar que apenas as denominações que estão no interior do símbolo são especificadas pela norma IEEE/ANSI. Os nomes que estão do lado de fora do símbolo não fazem parte do padrão, e, na verdade, eles variam de um fabricante de circuito integrado (CI) para outro.

Parte da notação utilizada na Fig. 7-26 já deve ser familiar para você. O contorno do símbolo está dividido em um bloco de controle comum, que afeta todos os flip-flops do contador, e quatro retângulos estreitos que representam os flip-flops individuais. O número entre parênteses, dentro de cada retângulo que representa um flip-flop, expressa seu peso no contador. O nome CTR DIV16 significa que este dispositivo, quando funciona normalmente, é um contador (CTR) com 16 estados (isto é, um divisor por 16). A entrada *MR* do bloco de controle comum tem a notação "CT = 0" para indicar que o contador irá para zero quando *MR* estiver em ALTO.

Dependência de Controle (C)

A letra C no nome de uma entrada indica que esta entrada *controla* a entrada de dados para o elemento armazenador (isto é, um flip-flop). Usualmente, C é utilizado para entradas que controlam a entrada de dados em um flip-flop na sua transição de disparo. Verificamos isto quando estuda-

*Esta seção pode ser omitida sem perda de continuidade.



Obs.: A entrada reset tem prioridade sobre as entradas $\overline{\text{Load}}$ e as entradas de clock.
A entrada $\overline{\text{Load}}$ tem prioridade sobre as entradas de clock.

Fig. 7-25 - Dois 74LS193s conectados em um arranjo de dois estágios para estender o intervalo máximo de contagem.

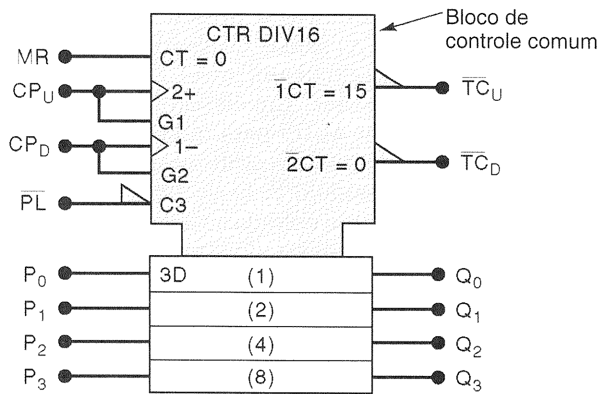


Fig. 7-26 - Símbolo IEEE/ANSI para o CI 74LS193.

mos os símbolos IEEE/ANSI para flip-flops no Cap. 5. Na Fig. 7-26, C é utilizado para a entrada de carga paralela $\overline{PL}$, uma vez que esta controla a entrada de dados nos quatro flip-flops do contador. Especificamente, a denominação C3 indica que esta entrada controlará qualquer outra entrada que possua o dígito 3 como um prefixo em seu nome. Neste caso, isto inclui as entradas P_0, P_1, P_2 e P_3, uma vez que todas elas têm a indicação 3D (isto é mostrado apenas no flip-flop superior, e consideramos que ele deve ser o mesmo para os outros flip-flops). O "D" presente na denominação se refere a "dados".

Isto significa que quando $\overline{PL}$ está em seu estado ativo BAIXO, dados presentes em P_0 a P_3 serão armazenados nos flip-flops Q_0 a Q_3. Uma vez que não há símbolo indicador de disparo por transição em $\overline{PL}$, está subentendido que $\overline{PL}$ afetará as saídas enquanto estiver no seu estado ativo BAIXO.

Sentido de Contagem (+ ou −)

As entradas CP_U e CP_D são mostradas na Fig. 7-26 como possuindo duas denominações separadas porque elas possuem vários efeitos internos distintos. Vamos considerar primeiro a denominação superior. Esta é 2+ para a entrada CP_U. O sinal de mais (+) indica que uma transição positiva nesta entrada vai incrementar o contador de 1; em outras palavras, vai fazer com que o contador conte de modo crescente. Do mesmo modo, a denominação superior para a entrada CP_D possui um sinal de menos (−) para mostrar que esta entrada decrementa o contador de 1; em outras palavras, vai causar uma contagem decrescente. O significado dos dígitos na frente dos sinais de mais e menos será explicado nos parágrafos seguintes.

Dependência AND (G)

A letra G na denominação para uma entrada representa uma dependência AND. Isto significa que é feita uma operação AND com uma entrada designada por G seguido por um dígito, e qualquer outra entrada ou saída que tenha este mesmo dígito como um prefixo da sua denominação. Na Fig. 7-26 vemos que a designação inferior para a entrada CP_U é G1. Isto significa que uma operação AND é internamente feita com CP_U e com qualquer entrada ou saída que tenha um 1 na sua denominação. A denominação superior para CP_D é 1−, e, portanto, deve haver uma dependência AND entre CP_D e CP_U. Especificamente, esta dependência AND nos diz que CP_U deve estar em ALTO para que CP_D possa realizar sua função de contagem decrescente.

A designação inferior para CP_D é G2. Isto significa que existe uma dependência AND entre CP_D e qualquer entrada ou saída que tenha um 2 na sua denominação. Por exemplo, a denominação superior para CP_U é 2+, que nos diz que CP_D deve estar em ALTO para que CP_U possa realizar sua função de contagem crescente.

Vamos agora observar a designação para a saída $\overline{TC}_D$. Ela é $\overline{2}CT = 0$. Ela inclui um 2 em sua designação, indicando que existe uma dependência AND com CP_D. Na verdade, como ele é um $\overline{2}$, a dependência AND é com $\overline{CP}_D$. Então, a denominação para $\overline{TC}_D$ nos diz que ele irá para o seu estado ativo em nível BAIXO quando $\overline{CP}_U$ estiver em BAIXO *e* o contador for igual a zero ($CT = 0$). De modo semelhante, a designação para $\overline{TC}_U$ nos diz que ele irá para o seu estado ativo em nível BAIXO quando $\overline{CP}_U$ estiver em BAIXO *e* a contagem for igual a 15 ($CT = 15$).

Questões de Revisão

1. Explique o significado da dependência de controle e da dependência AND.
2. Dê o significado das seguintes designações de entrada: **(a)** + **(b)** G4 **(c)** C5 **(d)** 5D

7-11 DECODIFICANDO UM CONTADOR

Contadores digitais são geralmente utilizados em aplicações onde a contagem representada pelo estado dos flip-flops deve ser de algum modo determinada ou visualizada. Uma das maneiras mais simples de visualizar o conteúdo de um contador é apenas conectar a saída de cada flip-flop a um LED [veja Fig. 7-5(b)]. Deste modo, os estados dos flip-flops são visivelmente representados pelos LEDs (aceso = 1, apagado = 0), e a contagem pode ser mentalmente determinada pela **decodificação** dos estados binários dos LEDs. Por exemplo, suponha que este método é usado para um contador BCD e os estados dos LEDs são apagado-aceso-aceso-apagado, respectivamente. Isto representaria 0110, que mentalmente decodificaríamos como o decimal 6. Outras combinações dos estados dos LEDs representariam as outras contagens possíveis.

O método que utiliza LEDs para visualização da contagem torna-se inconveniente à medida que o tamanho (número de bits) do contador aumenta, porque é muito mais difícil decodificar mentalmente os resultados mostrados. Por esta razão, seria desejável desenvolver um meio para *eletronicamente* decodificar o conteúdo de um contador e mostrar os resultados de uma forma que seria imediatamente reconhecida.

Existe uma razão ainda mais importante para a decodificação eletrônica do conteúdo de um contador: em muitas aplicações nas quais contadores são usados para controlar a

temporização ou o seqüenciamento das operações *automaticamente*, sem intervenção humana. Por exemplo, a operação de um certo sistema poderia ser iniciada quando o contador atingisse o estado 101100 (contagem de 44_{10}). Um circuito lógico pode ser usado para decodificar ou para detectar quando esta contagem em particular estiver presente e então iniciar a operação. Muitas operações têm que ser controladas desta maneira em um sistema digital. Obviamente, a intervenção humana neste processo seria indesejável, a não ser que o sistema fosse extremamente lento.

Decodificação Ativa em ALTO

Um contador de módulo X possui X estados diferentes. Cada estado é uma seqüência particular de 0s e 1s armazenados nos flip-flops do contador. Uma malha de decodificação é um circuito lógico que gera X saídas diferentes, cada uma das quais detecta (decodifica) a presença de um estado particular do contador. As saídas do decodificador podem ser projetadas para produzir um nível ALTO ou BAIXO quando a detecção ocorrer. Um decodificador ativo em ALTO produz saídas em ALTO para indicar a detecção. A Fig. 7-27 mostra um circuito decodificador ativo em ALTO para um contador de módulo 8. O decodificador consiste em oito portas AND de três entradas. Cada porta AND produz um nível ALTO para um estado particular do contador.

Por exemplo, a porta AND 0 tem em suas entradas as saídas dos flip-flops $\overline{C}$, $\overline{B}$ e $\overline{A}$. Então, sua saída estará em BAIXO durante todo o tempo, *exceto* quando $A = B = C = 0$, isto é, na contagem de 000 (zero). De modo similar, a porta AND 5 tem em suas entradas as saídas dos flip-flops C, $\overline{B}$ e A, e portanto sua saída irá para ALTO apenas quando $C = 1$, $B = 0$ e $A = 1$, isto é, na contagem de 101 (5 decimal). O resto das portas AND opera de modo semelhante

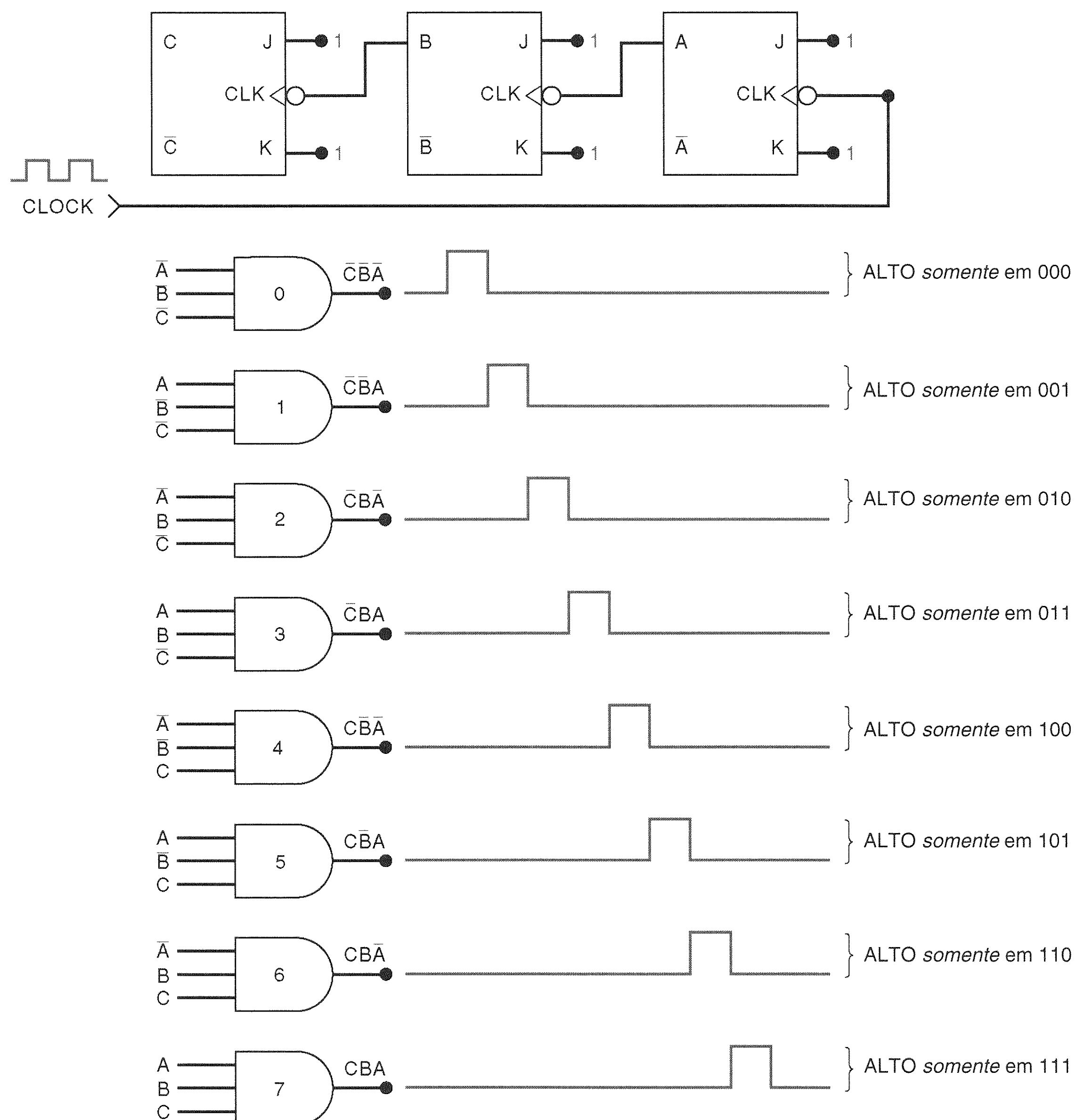


Fig. 7-27 - Usando portas AND para decodificar um contador de módulo 8.

para as outras contagens possíveis. Em qualquer instante de tempo, apenas a saída de uma única porta AND estará em ALTO, aquela que está decodificando a contagem em particular que está presente no contador. As formas de onda na Fig. 7-27 mostram isto claramente.

As oito saídas das portas AND podem ser usadas para controlar oito LEDs, representando números decimais de 0 a 7. Apenas um LED estará aceso em cada instante de tempo, indicando a contagem apropriada.

O decodificador feito com portas AND pode ser estendido a contadores com qualquer número de estados. O exemplo a seguir ilustra esta afirmação.

EXEMPLO 7-16

Quantas portas AND são necessárias para decodificar completamente todos os estados de um contador binário de módulo 32? Quais são as entradas da porta que detecta a contagem 21?

Solução

Um contador de módulo 32 possui 32 estados possíveis. Uma porta AND é necessária para cada estado, e, portanto, o decodificador necessitará de 32 portas AND. Uma vez que $32 = 2^5$, o contador possui cinco flip-flops. Portanto, cada porta terá cinco entradas, uma de cada flip-flop. Para decodificar a contagem de 21 (isto é, 10101_2) necessita-se que as entradas da porta AND sejam E, $\overline{D}$, C, $\overline{B}$ e A, onde E é o flip-flop mais significativo.

Decodificação Ativa em BAIXO

Se usarmos portas NAND no lugar das AND, as saídas do decodificador produzirão normalmente um sinal ALTO, que irá para BAIXO apenas quando o número que está sendo decodificado ocorrer. Ambos os tipos de decodificadores são usados, dependendo do tipo de circuitos que estão sendo acionados pelas saídas do decodificador.

EXEMPLO 7-17

A Fig. 7-28 mostra uma situação comum, na qual um contador é usado para gerar uma forma de onda que poderia ser utilizada para controlar dispositivos tais como um motor, uma válvula solenóide ou um aquecedor. Um contador de módulo 16 passa por sua seqüência de contagem continuamente. Cada vez que ele atingir a contagem de 8 (1000), a porta NAND superior produzirá uma saída em BAIXO, que coloca o flip-flop X no estado 1. O flip-flop X permanece em ALTO até que o contador atinja o estado 14 (1110); neste instante, a porta NAND inferior decodifica este estado e produz uma saída em BAIXO, que coloca X no estado 0. Então a saída X estará em ALTO entre as contagens de 8 e 14 para cada ciclo do contador.

Decodificação de um Contador BCD

Um contador BCD possui 10 estados, que podem ser decodificados utilizando-se as técnicas descritas anteriormente. Decodificadores BCD fornecem 10 saídas que correspondem aos dígitos decimais 0 a 9 representados pelos estados dos flip-flops do contador. Estas 10 saídas podem ser usadas para controlar 10 LEDs e fornecer uma indicação visual. Mais freqüentemente, em vez de 10 LEDs, um display é utilizado para mostrar números decimais de 0 a 9. Um dispositivo deste tipo é o chamado *nixie tube*, que contém 10 filamentos muito finos em forma de números colocados uns sobre os outros. As saídas do decodificador BCD controlam

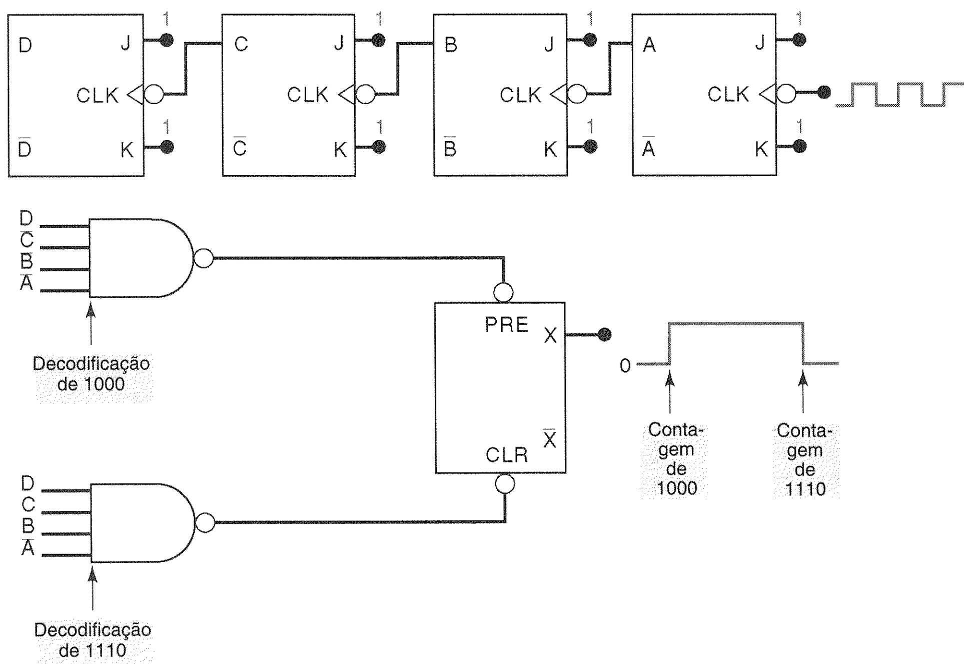


Fig. 7-28 - Exemplo 7-17.

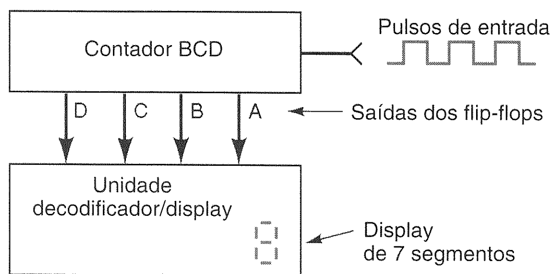


Fig. 7-29 - Contadores BCD geralmente têm sua contagem mostrada em um único display.

que filamento deve ser iluminado. Uma nova classe de displays decimais contém sete pequenos segmentos feitos de um material (geralmente LEDs ou displays de cristal líquido) que ou emite luz ou reflete a luz ambiente. As saídas do decodificador BCD controlam que segmentos são iluminados, de modo a produzir um padrão que represente um dos dígitos decimais.

Entraremos em maiores detalhes sobre estes tipos de decodificadores e displays no Cap. 9. Entretanto, uma vez que contadores BCD e seus decodificadores e displays associados são muito comuns, usaremos uma unidade decodificador/display (veja Fig. 7-29) para representar o circuito completo usado para apresentar o conteúdo de um contador BCD como um dígito decimal.

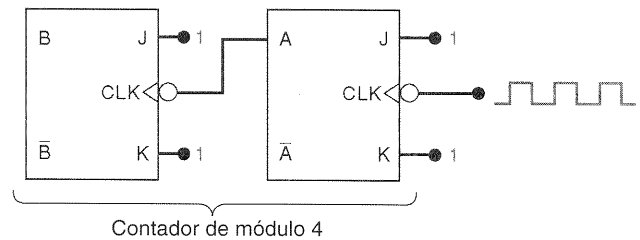
Questões de Revisão

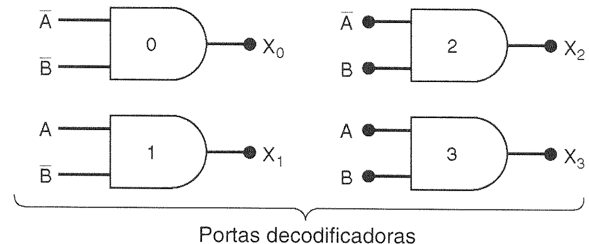
1. Quantas portas são necessárias para decodificar completamente um contador de 6 bits?
2. Descreva a porta decodificadora necessária para produzir uma saída em BAIXO quando um contador de módulo 64 está na contagem de 23.

7-12 GLITCHES DE DECODIFICAÇÃO

Na Seção 7-5 discutimos os efeitos dos atrasos de propagação em contadores assíncronos. Como foi visto naquela seção, os atrasos de propagação acumulados vão limitar a freqüência máxima deste tipo de contador. Os atrasos entre as transições dos flip-flops podem causar problemas quando estivermos decodificando um contador assíncrono. O problema aparece na forma de **glitches de decodificação**, isto é, pulsos estreitos que aparecem nas saídas de algumas portas decodificadoras. Isto é ilustrado na Fig. 7-30 para um contador assíncrono de módulo quatro.

As formas de onda nas saídas de cada flip-flop e de cada porta decodificadora podem ser vistas na figura. Observe o atraso de propagação entre o sinal de clock e a saída A e aquele existente entre as formas de onda de A e B. Os glitches em X_0 e X_2 são causados pelo atraso de propagação entre as formas de onda A e B. X_0 é a saída da porta AND decodificadora para a contagem 00. Esta condição 00 também ocorre momentaneamente quando o contador vai da contagem 01 para 10, como pode ser visto nas formas de onda. Isto acontece porque B não pode mudar de estado até que A vá para BAIXO. Esse estado momentâneo 00 dura vários nanossegundos (dependendo do t_{pd} do flip-flop B), mas pode ser detectado pela porta decodificadora caso a resposta desta porta seja suficientemente rápida. Daí o pulso estreito na saída X_0.





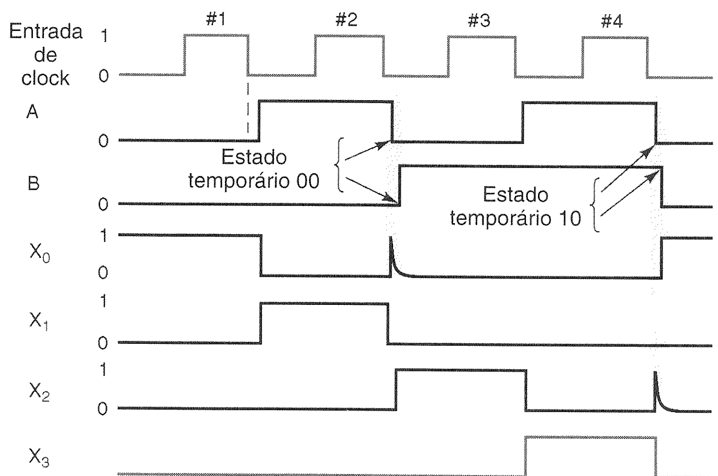


Fig. 7-30 - As formas de onda nos flip-flops e as formas de onda decodificadas para um contador assíncrono de módulo 4, mostrando glitches nas saídas X_0 e X_2.

Uma situação similar produz um glitch na saída X_2. X_2 é o resultado da decodificação da condição 10, que ocorre momentaneamente quando o contador vai de 11 para 00 em resposta ao quarto pulso de clock, como pode ser visto nas formas de onda. Novamente, isto acontece devido ao atraso na resposta do flip-flop B após o sinal A ter ido para BAIXO.

Embora esta situação esteja ilustrada para um contador de módulo 4, este mesmo tipo de situação pode ocorrer para *qualquer* contador por pulsação. Isto acontece porque este tipo de contador trabalha de acordo com o princípio da "reação em cadeia", onde cada flip-flop dispara o seguinte e assim por diante. Os pulsos estreitos nas saídas decodificadas podem ou não representar um problema, dependendo do modo como o contador está sendo usado. Quando o con-

tador está sendo usado apenas para contar pulsos e mostrar os resultados, os glitches decodificados não representam maiores problemas, pois estes são de duração muito curta e nem serão mostrados no display. Entretanto, quando o contador é usado para controlar outros circuitos lógicos, como foi feito na Fig. 7-28, estes pulsos estreitos podem causar uma operação imprópria. Por exemplo, um pulso estreito na saída de uma das portas NAND decodificadora faria com que o flip-flop fosse para o estado 1 ou para o estado 0 no momento errado.

Podemos prever que ocorrerá um estado temporário em uma seqüência de contagem assíncrona acompanhando uma transição no estado de contagem em um flip-flop de cada vez. Por exemplo, vamos observar passo a passo o processo pelo qual o contador vai de 011 (3) para 100 (4):

	C	*B*	*A*		
	0	1	1	(3)	
estados temporários	0	1	0	(2)	< FF *A* comuta primeiro
	0	0	0	(0)	< e faz com que *B* comute
	1	0	0	(4)	< o que faz com que *C* comute.

Observe a ocorrência de dois estados temporários: 010 e 000.

Nas situações onde pulsos estreitos decodificados não podem ser tolerados, existem duas soluções básicas para o problema. A primeira é usar um contador síncrono em vez de um contador assíncrono. Lembre-se de que em contadores síncronos todos os flip-flops são disparados ao mesmo tempo pelo pulso de clock, de modo que parece que as condições que produziram pulsos estreitos não podem ocorrer. Entretanto, mesmo em um contador síncrono estes pulsos estreitos podem ser produzidos porque não necessariamente todos os flip-flops têm o mesmo t_{pd}, especialmente quando alguns flip-flops tiverem suas saídas mais carregadas do que outros.

Strobing (Amostragem)

O método mais confiável de eliminar estes pulsos estreitos decodificados é usar uma técnica chamada **strobing**. Esta técnica utiliza um sinal chamado *strobe* que mantém as portas AND decodificadoras desabilitadas (saídas em 0) até que todos os flip-flops tenham atingido um estado estável em resposta à transição negativa do clock. Isto está ilustrado na Fig. 7-31, onde o sinal de strobe é conectado como uma entrada em cada uma das portas decodificadoras. As formas de onda mostram que o sinal de strobe vai para BAIXO quando o pulso de clock vai para ALTO. Durante o tempo em que o strobe está BAIXO, as portas decodificadoras são mantidas em BAIXO. O sinal de strobe vai para ALTO, para habilitar as portas decodificadoras, algum tempo t_D *depois* de o pulso de clock ir para BAIXO. t_D é escolhido para ser maior que o intervalo total necessário para que o contador atinja uma contagem estável, e isto depende, é claro, dos atrasos de propagação e do número de flip-flops do contador. Neste método, as saídas das portas decodificadoras não terão nenhum pulso estreito, porque estarão desabilitadas durante o tempo em que os flip-flops estiverem em transição.

O método de strobe não é utilizado se o contador é usado apenas para visualização, uma vez que os pulsos decodificados são muito estreitos para afetar a apresentação. O sinal de strobe é usado quando o contador é utilizado em aplicações de controle, como a da Fig. 7-28, onde estes pulsos poderiam causar operação incorreta.

Questões de Revisão

1. Explique por que as portas decodificadoras para um contador assíncrono podem ter glitches nas suas saídas.
2. Como a amostragem elimina os glitches decodificados?

7-13 LIGAÇÃO EM CASCATA DE CONTADORES BCD

Contadores BCD são freqüentemente usados quando pulsos devem ser contados e o resultado deve ser mostrado

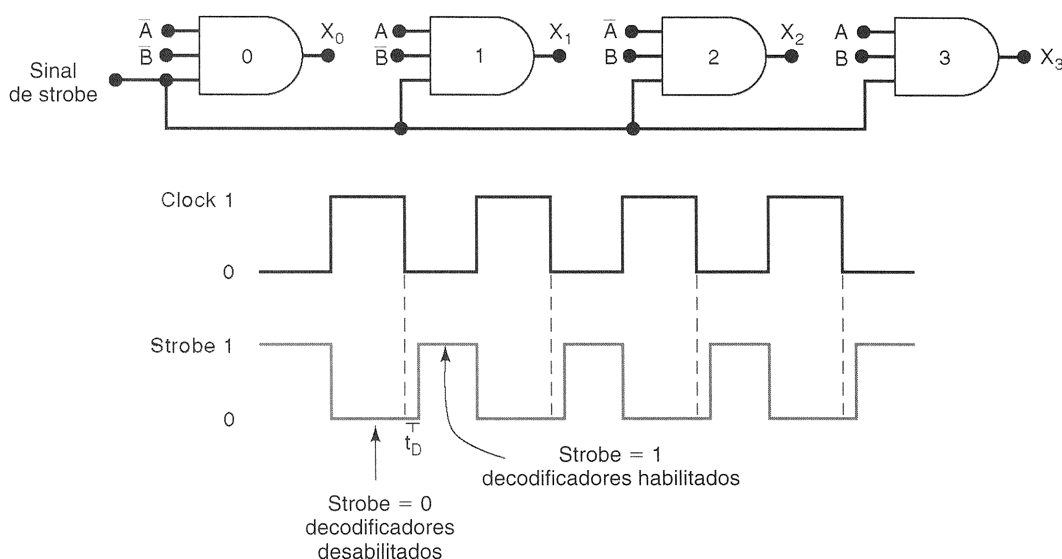


Fig. 7-31 - Uso do sinal de strobe para eliminar spikes de decodificação.

em decimal. Um contador BCD simples conta de 0 a 9 e depois retorna a 0. Para contar valores decimais maiores, podemos **ligar em cascata** estágios contadores BCD, como pode ser visto na Fig. 7-32. Este arranjo com vários estágios opera da seguinte maneira:

1. Inicialmente, todos os contadores são colocados no estado 0. Portanto, é mostrado 000.
2. À medida que os pulsos de entrada chegam, o contador BCD das unidades avança uma contagem por pulso. Após terem ocorrido nove pulsos, os contadores BCD das dezenas e das centenas ainda estão em 0, e o contador das unidades está em 9 (binário 1001). Então, o número mostrado é 009.
3. No décimo pulso de entrada, o contador das unidades retorna a 0 fazendo com que a saída do seu flip-flop *D* vá de 1 para 0. Esta transição negativa age como uma entrada de clock para o contador das dezenas, fazendo com que este avance uma contagem. Então, após 10 pulsos, o número lido será 010.
4. À medida que pulsos adicionais ocorrem, o contador das unidades avança uma contagem por pulso, e toda vez que este retorna a 0, ele faz com que o contador das dezenas avance uma contagem. Então, após 99 pulsos de entrada terem ocorrido, o contador das dezenas está em 9, assim como o contador das unidades. O número decimal lido é, portanto, 099.
5. No centésimo pulso de entrada, o contador de unidades retorna a 0, o que faz com que o contador das dezenas retorne a 0. A saída do flip-flop do contador das dezenas faz uma transição negativa que age como um clock para o contador das centenas e faz com que este avance a contagem. Então, após 100 pulsos, o número lido é igual a 100.
6. Este processo continua até que tenham ocorrido 999 pulsos de entrada. No milésimo pulso, todos os contadores retornam a 0.

Deveria ser óbvio que este arranjo pode ser expandido para um número qualquer de dígitos decimais através da simples adição de mais estágios. Por exemplo, para contar até 999.999 serão necessários seis contadores BCD e seus decodificadores e displays associados. De um modo geral, precisamos de um contador BCD para cada dígito decimal.

Cada contador BCD neste arranjo em cascata, como aquele da Fig. 7-32, poderia ser um contador de módulo variável como o 74LS293 configurado como um contador de módulo 10, ou poderia ser um CI que é internamente ligado como um contador BCD como os CIs 74LS90 e 74LS192/HC192.

7-14 PROJETO DE CONTADORES SÍNCRONOS*

Vários tipos de contadores estão disponíveis na forma de circuitos integrados: assíncronos, síncronos e síncronos/assíncronos. A maioria deles conta segundo uma seqüência binária normal, embora suas seqüências de contagem possam ser de algum modo alteradas usando os métodos demonstrados para os CIs 74293 e 74193. Existem situações, entretanto, em que um contador deve seguir uma seqüência que não é aquela binária normal, como por exemplo 000, 010, 101, 001, 110, 000 ...

Existem vários métodos para projetar contadores que sigam seqüências arbitrárias. Apresentaremos em detalhe um método muito comum que utiliza flip-flops J-K em arranjos contadores síncronos. Este mesmo método pode ser usado em projetos com o flip-flop D. Esta técnica é um dos vários procedimentos de projeto que fazem parte de uma área de projeto de circuitos digitais chamada **projeto de circuitos seqüenciais**, que normalmente faz parte de um curso avançado.

Idéia Básica

Em contadores síncronos, todos os flip-flops são disparados ao mesmo tempo. Antes de cada pulso de clock, as entradas *J* e *K* de cada flip-flop devem estar no nível correto para garantir que o flip-flop vá para o estado correto. Por exemplo, considere a situação mostrada na Tabela 7-1. Quando ocorrer o próximo pulso de clock, as entradas *J* e *K* dos flip-flops devem estar nos níveis corretos para fazer com que o flip-flop *C* mude de 1 para 0, o flip-flop *B* de 0 para 1 e o flip-flop *A* de 1 para 1 (isto é, não mude).

O procedimento para projetar um contador síncrono torna-se um processo de projeto de circuitos lógicos, que *decodificam* os vários estados do contador para fornecer os níveis lógicos para cada entrada *J* e *K*. As entradas destes

*Esta seção pode ser omitida sem afetar a continuidade do restante do livro.

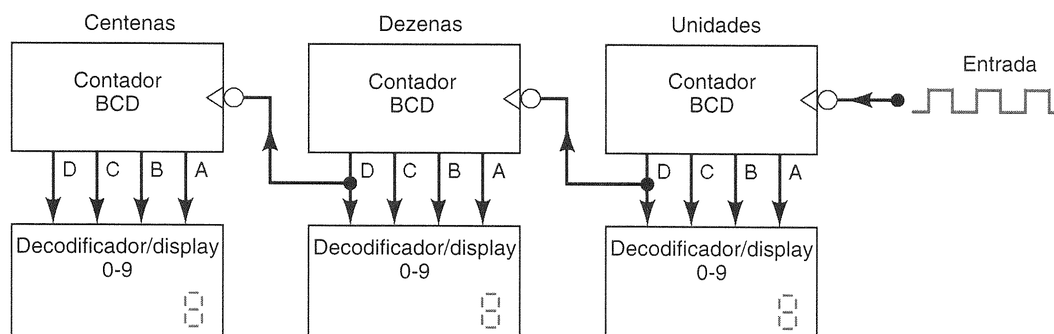


Fig. 7-32 - Contadores BCD em cascata para contar e mostrar números de 000 a 999.

TABELA 7-1

Estado ATUAL			PRÓXIMO Estado		
C	*B*	*A*	*C*	*B*	*A*
1	0	1	0	1	1

circuitos decodificadores são provenientes das saídas de um ou mais flip-flops. Para exemplificar, para o contador síncrono da Fig. 7-17, a porta AND, que fornece as entradas *J* e *K* do flip-flop *C*, decodifica os estados dos flip-flops *A* e *B*. Do mesmo modo, a porta AND, que fornece as entradas *J* e *K* do flip-flop *D*, decodifica os estados dos flip-flops *A*, *B* e *C*.

Tabela de Excitação J-K

Antes de iniciarmos o processo de projetar circuitos decodificadores para cada entrada *J* e *K*, devemos primeiro rever a operação de um flip-flop J-K usando uma abordagem diferente através da *tabela de excitação* (Tabela 7-2). A coluna mais à esquerda desta tabela enumera cada transição possível da saída de um flip-flop. A segunda e terceira colunas relacionam o estado atual do flip-flop, simbolizado por $Q(N)$, e o próximo estado simbolizado por $Q(N + 1)$, para cada transição. As duas últimas colunas enumeram os níveis lógicos nas entradas *J* e *K* necessários para produzir cada uma das transições. Vamos examinar cada caso.

TRANSIÇÃO 0 → 0 O estado atual do flip-flop é 0 e ele deve permanecer em 0 quando o pulso de clock for aplicado. A partir da nossa compreensão de como um flip-flop J-K funciona, isto pode acontecer quando ou $J = K = 0$ (condição sem mudança) ou $J = 0$ e $K = 1$ (condição de reset). Portanto, *J* deve estar em 0, mas *K* pode estar em qualquer nível. A tabela indica este fato com um "0" em *J* e um "*x*" em *K*. Lembre-se de que "*x*" representa uma condição "*don't care*".

TRANSIÇÃO 0 → 1 O estado atual é 0 e deve mudar para 1. Isto pode acontecer quando $J = 1$ e $K = 0$ (condição set) ou $J = K = 1$ (condição de comutação). Portanto, *J* deve estar em nível 1, mas *K* pode estar em qualquer nível para esta transição ocorrer.

TRANSIÇÃO 1 → 0 O estado atual é 1 e deve mudar para 0. Isto pode acontecer quando ou $J = 0$ e $K = 1$ ou $J = K = 1$. Portanto, *K* deve estar em 1, mas *J* pode estar em qualquer nível.

TABELA 7-2 Tabela de excitação do flip-flop J-K.

Transição na Saída	Estado ATUAL $Q(N)$	PRÓXIMO Estado $Q(N + 1)$	*J*	*K*
0 → 0	0	0	0	*x*
0 → 1	0	1	1	*x*
1 → 0	1	0	*x*	1
1 → 1	1	1	*x*	0

TRANSIÇÃO 1 → 1 O estado atual é 1 e deve permanecer em 1. Isto pode ocorrer quando $J = K = 0$ ou $J = 1$ e $K = 0$. Portanto, *K* deve estar em 0, enquanto *J* pode estar em qualquer nível.

O uso desta **tabela de excitação J-K** (Tabela 7-2) é a parte principal do procedimento de projeto de contadores síncronos.

Procedimento de Projeto

Passaremos agora por um procedimento completo de projeto de contadores síncronos. Embora façamos isso para uma seqüência de contagem específica, os mesmos passos podem ser aplicados para qualquer seqüência desejada.

Passo 1. Determine o número de bits necessários (número de flip-flops) e a seqüência de contagem desejada.

Para o nosso exemplo, projetaremos um contador de três bits cuja seqüência de contagem pode ser vista na Tabela 7-3. Observe que esta seqüência não inclui os estados 101, 110 e 111. Vamos nos referir a eles como *estados indesejáveis*.

Passo 2. Desenhe o diagrama de transição de estados mostrando **todos** os estados possíveis, incluindo aqueles que não fazem parte da seqüência de contagem desejada.

Para o nosso exemplo, o diagrama de transição de estados pode ser visto na Fig. 7-33. Os estados 000 a 100 estão ligados segundo a seqüência esperada. O que há de novo neste diagrama é a inclusão dos estados indesejáveis. Eles devem ser incluídos em nosso projeto para o caso de o contador ir para um desses estados ao ligar o circuito ou devido ao ruído presente. O projetista pode escolher para cada um dos estados indesejáveis para qual estado ele deve ir mediante aplicação do próximo pulso de clock. Escolhemos que todos eles devem ir para o estado 000 a partir do qual a seqüência correta de contagem será gerada.

Passo 3. Use o diagrama de transição de estados para construir uma tabela que relacione **todos** os estados ATUAIS e seus PRÓXIMOS estados.

Para o nosso exemplo, esta informação pode ser vista na Tabela 7-4. O lado esquerdo da tabela relaciona *todos* os estados possíveis, mesmo aqueles que não fazem parte da seqüência. Vamos denominá-los estados ATUAIS. O lado direito enumera o PRÓXIMO estado para cada estado ATU-

TABELA 7-3

C	*B*	*A*
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
0	0	0
0	0	1
	etc.	

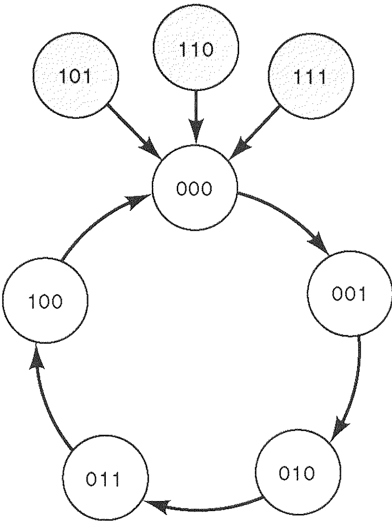


Fig.7-33 - Diagrama de transição de estados para o exemplo de projeto do contador síncrono.

TABELA 7-4

		Estado ATUAL			PRÓXIMO Estado		
		C	*B*	*A*	*C*	*B*	*A*
linha	1	0	0	0	0	0	1
	2	0	0	1	0	1	0
	3	0	1	0	0	1	1
	4	0	1	1	1	0	0
	5	1	0	0	0	0	0
	6	1	0	1	0	0	0
	7	1	1	0	0	0	0
	8	1	1	1	0	0	0

AL. Estes podem ser obtidos a partir do diagrama de transição de estados da Fig. 7-33. Por exemplo, a linha 1 mostra que o estado ATUAL 000 tem como PRÓXIMO estado 001, a linha 5 mostra que o estado ATUAL 100 tem como PRÓXIMO estado 000. As linhas 6, 7 e 8 mostram que os estados ATUAIS indesejáveis 101, 110 e 111 têm como PRÓXIMO estado 000.

Passo 4. Acrescente uma coluna a esta tabela para cada entrada *J* e *K*. Para cada estado ATUAL, indique os níveis necessários em cada entrada *J* e *K* para produzir a transição para o PRÓXIMO estado.

Nosso exemplo utiliza três flip-flops — *C*, *B* e *A* — e cada um deles tem entradas *J* e *K*. Portanto, devemos adicionar seis novas colunas como mostrado na Tabela 7-5. Esta tabela completa é chamada de **tabela de excitação do circuito**. As seis novas colunas são as entradas *J* e *K* de cada flip-flop. Os valores para cada coluna *J* e *K* são obtidos utilizando a Tabela 7-2, que é a tabela de excitação do flip-flop J-K que desenvolvemos anteriormente. Demonstraremos isto para vários casos, e você pode verificar o resto.

Vamos observar a linha 1 da Tabela 7-5. O estado ATUAL 000 deve ir para o PRÓXIMO estado 001 na ocorrência de um pulso de clock. Para esta transição de estados, o flip-flop *C* vai de 0 para 0. Pela tabela de excitação J-K, veremos que J_C deve estar em 0 e K_C em "x" para que esta transição ocorra. O flip-flop *B* vai de 0 para 0 e, portanto, $J_B = 0$ e $K_B = x$. O flip-flop *A* vai de 0 para 1. Também a partir da Tabela 7-2, vemos que $J_A = 1$ e $K_A = x$ para esta transição.

Na linha 4 da Tabela 7-5, o estado ATUAL 011 tem como PRÓXIMO estado 100. Para esta transição de estado, o flip-flop *C* vai de 0 para 1, o que requer que $J_C = 1$ e $K_C = x$. Os flip-flops *A* e *B* estão ambos indo de 1 para 0. A tabela de excitação do J-K indica que estes dois flip-flops necessitam de que $J = x$ e $K = 1$ para que isto ocorra.

Os níveis necessários para todas as outras linhas da Tabela 7-5 podem ser determinados da mesma maneira.

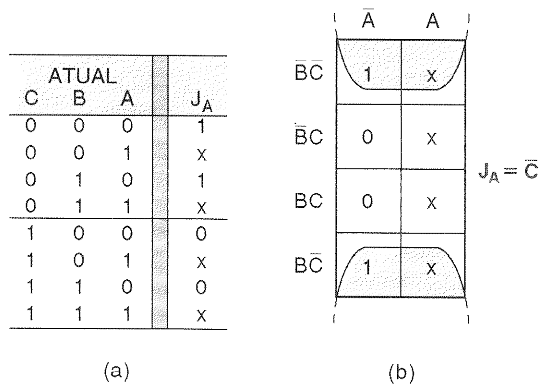
Passo 5. Projete os circuitos lógicos que forneçam os níveis necessários para cada entrada *J* e *K*.

A Tabela 7-5, que é a tabela de excitação do circuito, relaciona as seis entradas *J* e *K*: J_C, K_C, J_B, K_B, J_A e K_A. Devemos considerar cada uma destas entradas como saídas de um circuito lógico próprio cujas entradas são provenientes dos flip-flops *C*, *B* e *A*. Portanto, devemos projetar um circuito lógico para cada uma destas entradas. Vamos projetar o circuito para J_A.

Para fazer isto, devemos observar o estado ATUAL presente nos flip-flops *C*, *B* e *A* e os níveis desejados para J_A em cada caso. Esta informação pode ser obtida da Tabela 7-5 e pode ser vista na Fig. 7-34(a). Esta tabela-verdade mostra os níveis desejados para J_A, para cada estado ATUAL. É claro que para alguns destes casos temos uma condição "*don't care*" para J_A. Para obter o circuito lógico para J_A, devemos determinar sua expressão em termos de *C*, *B* e *A*.

TABELA 7-5 Tabela de excitação do circuito.

		Estado ATUAL			PRÓXIMO Estado								
		C	*B*	*A*	*C*	*B*	*A*	J_C	K_C	J_B	K_B	J_A	K_A
linha	1	0	0	0	0	0	1	0	x	0	x	1	x
	2	0	0	1	0	1	0	0	x	1	x	x	1
	3	0	1	0	0	1	1	0	x	x	0	1	x
	4	0	1	1	1	0	0	1	x	x	1	x	1
	5	1	0	0	0	0	0	x	1	0	x	0	x
	6	1	0	1	0	0	0	x	1	0	x	x	1
	7	1	1	0	0	0	0	x	1	x	1	0	x
	8	1	1	1	0	0	0	x	1	x	1	x	1



ATUAL C	B	A	J_A
0	0	0	1
0	0	1	x
0	1	0	1
0	1	1	x
1	0	0	0
1	0	1	x
1	1	0	0
1	1	1	x

Fig. 7-34 - (a) Porção da tabela de excitação do circuito mostrando J_A para cada estado ATUAL; (b) mapa de Karnaugh usado para obter uma expressão simplificada para J_A.

Faremos isto transferindo a informação contida na tabela-verdade para um mapa de Karnaugh de três variáveis e realizando a simplificação como mostra a Fig. 7-34(b).

Existem apenas dois 1s neste mapa de Karnaugh, que podem ser agrupados para obter o termo $\overline{A}\,\overline{C}$, mas se utilizarmos as condições *don't care* $A\overline{B}\,\overline{C}$ e $AB\overline{C}$ como 1s, podemos agrupar um quarteto para obter o termo mais simples $\overline{C}$. Portanto, a expressão final é

$$J_A = \overline{C}$$

Agora, vamos considerar K_A. Podemos seguir os mesmos passos que fizemos para J_A. Entretanto, observando os valores de K_A na tabela de excitação do circuito, temos apenas 1s e condições do tipo *don't care*. Se trocarmos todas as condições *don't care* por 1s, teremos K_A sempre igual a 1. Portanto, a expressão final é

$$K_A = 1$$

De uma maneira similar, podemos obter expressões para J_C, K_C, J_B e K_B. Os mapas de Karnaugh para estas expressões podem ser vistos na Fig. 7-35. Você pode querer confirmar se as expressões estão corretas conferindo-as com a tabela de excitação do circuito.

Passo 6. Implemente as expressões finais.

Os circuitos lógicos para cada entrada J e K são implementados a partir das expressões obtidas no mapa de Karnaugh. O circuito completo do contador síncrono projetado está na Fig. 7-36. Observe que todos os flip-flops são disparados pelo mesmo sinal de clock. Você pode querer

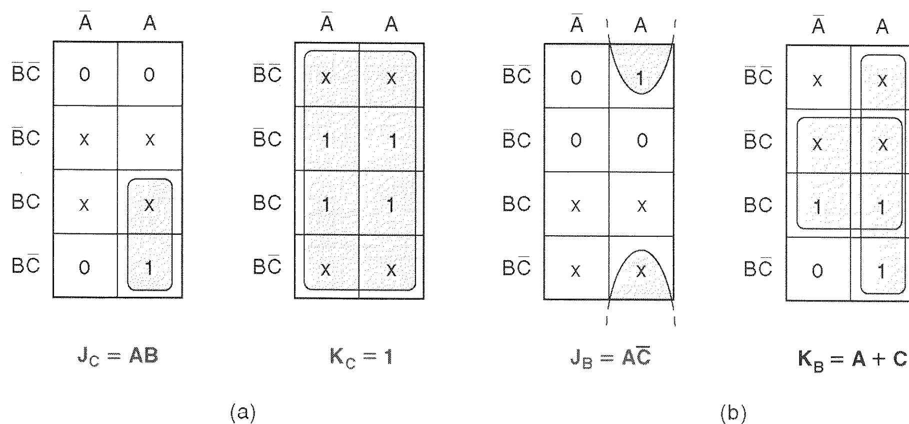


Fig. 7-35 - (a) Mapas de Karnaugh para os circuitos lógicos J_C e K_C; (b) mapas de Karnaugh para os circuitos lógicos J_B e K_B.

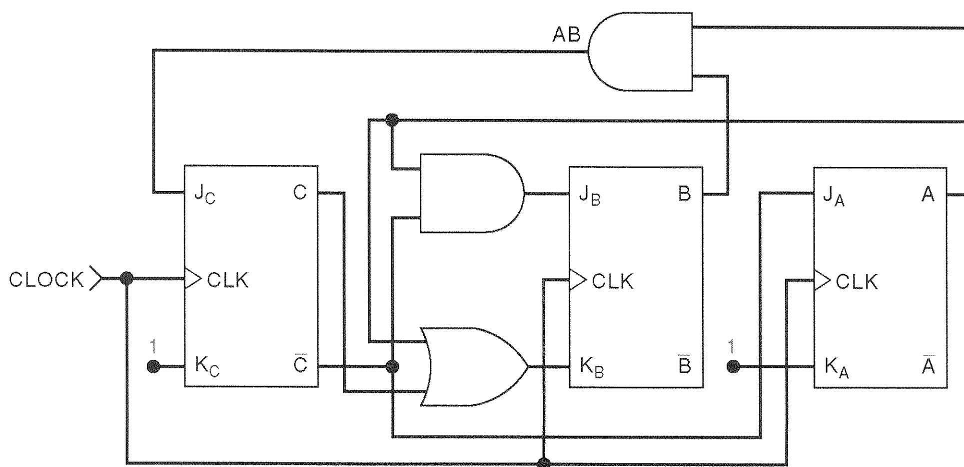


Fig. 7-36 - Implementação final do exemplo de projeto de um contador síncrono.

verificar que os circuitos lógicos para as entradas *J* e *K* concordam com as Figs. 7-34 e 7-35.

Controle de um Motor de Passo

Vamos aplicar este procedimento de projeto em uma situação prática — o controle de um *motor de passo*. Um motor de passo gira em passos discretos, geralmente 15° por passo, em vez de girar em movimento contínuo. Os enrolamentos dentro do motor devem ser energizados e desenergizados em uma seqüência específica para produzir movimento em passos discretos. Sinais digitais são normalmente usados para controlar a corrente em cada enrolamento do motor. Motores de passo são bastante utilizados em situações onde o controle preciso de posição é necessário, como por exemplo no posicionamento de cabeças para leitura/escrita de discos magnéticos, no controle de cabeças de impressão em impressoras e em robôs.

A Fig. 7-37(a) mostra um diagrama de um típico motor de passo de quatro enrolamentos. Para que o motor gire de modo correto, os enrolamentos 1 e 2 devem estar sempre em estados opostos, isto é, quando o enrolamento 1 está energizado, o enrolamento 2 não está, e vice-versa. Do mesmo modo, os enrolamentos 3 e 4 devem estar sempre em estados opostos. As saídas de um contador síncrono de dois bits são usadas para controlar a corrente nos quatro enrolamentos. A e $\overline{A}$ controlam os enrolamentos 1 e 2, e B e $\overline{B}$ controlam os enrolamentos 3 e 4. Amplificadores de corrente são necessários porque as saídas dos flip-flops não podem gerar a corrente exigida pelos enrolamentos.

Uma vez que o motor de passo pode girar em sentido horário ou anti-horário, temos uma entrada *D* que é usada para controlar a direção de rotação. Os diagramas de estado para as duas situações podem ser vistos na Fig. 7-37(b). Para termos a rotação em sentido horário, devemos ter $D = 0$, e o estado do contador *BA* deve seguir a seqüência 11, 10, 00, 01, 11, 10, ..., e assim sucessivamente, à medida que ocorra um pulso na entrada Passo. Para a rotação em sentido anti-horário, temos que *D* deve ser igual a 1 e que o contador deve seguir a seqüência 11, 01, 00, 10, 11, 01, ... e assim por diante.

Estamos agora prontos para seguir os seis passos para o projeto de um contador síncrono. Os passos 1 e 2 já foram feitos, e podemos proceder com os passos 3 e 4. A Tabela 7-6 mostra cada estado ATUAL possível para *D*, *B* e *A* e o próximo estado desejado, juntamente com os níveis lógicos para as entradas *J* e *K* necessários para alcançar todas as transições. Observe que em todos os casos, a entrada que indica a direção, *D*, não muda do estado ATUAL para PRÓXIMO; isto acontece porque ela é uma entrada independente que é mantida em ALTO ou BAIXO à medida que o contador avança em sua seqüência.

O passo 5 do procedimento de projeto é apresentado na Fig. 7-38, onde a informação na Tabela 7-6 foi transferida para os mapas de Karnaugh que mostram como cada sinal *J* e *K* está relacionado com o estado ATUAL de *D*, *B* e *A*. Fazendo os agrupamentos apropriados, as expressões lógicas simplificadas para cada sinal são obtidas.

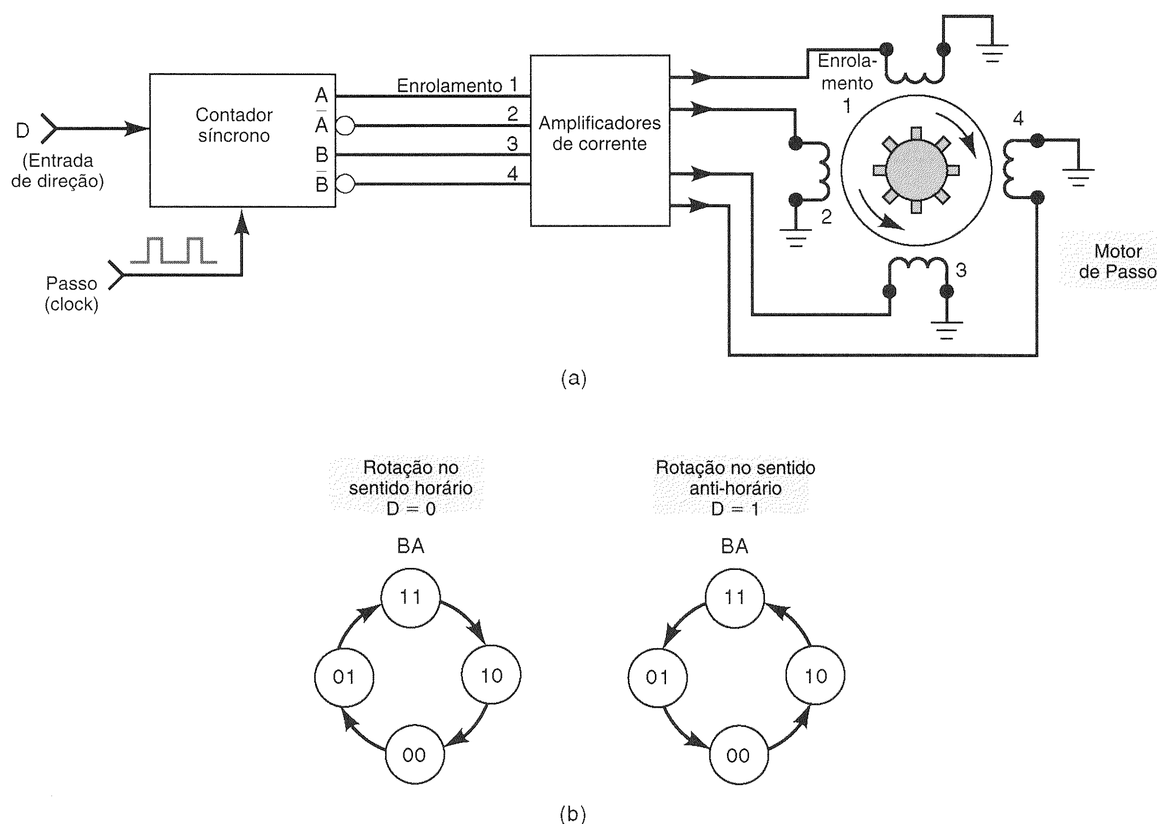
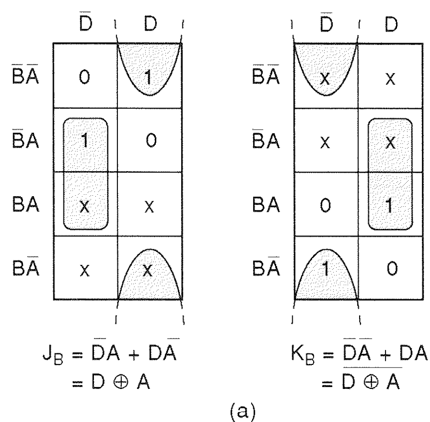


Fig. 7-37- (a) Um contador síncrono fornece a seqüência apropriada de saídas para acionar o motor de passo; (b) diagramas de transição de estados para os dois valores da entrada de direção *D*.

TABELA 7-6 Tabela de excitação do circuito da Fig. 7-37(b).

Estado ATUAL			PRÓXIMO Estado						
D	*B*	*A*	*D*	*B*	*A*	J_B	K_B	J_A	K_A
0	0	0	0	0	1	0	*x*	1	*x*
0	0	1	0	1	1	1	*x*	*x*	0
0	1	0	0	0	0	*x*	1	0	*x*
0	1	1	0	1	0	*x*	0	*x*	1
1	0	0	1	1	0	1	*x*	0	*x*
1	0	1	1	0	0	0	*x*	*x*	1
1	1	0	1	1	1	*x*	0	1	*x*
1	1	1	1	0	1	*x*	1	*x*	0



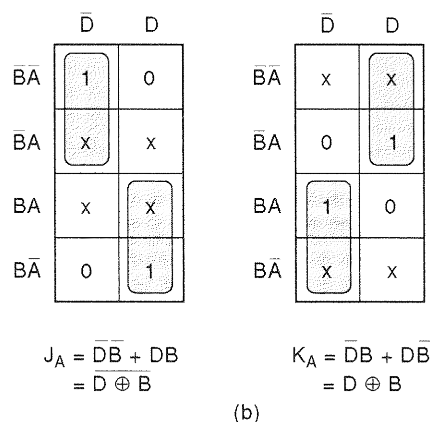


Fig. 7-38 - (a) Mapas de Karnaugh para J_B e K_B; (b) mapas de Karnaugh para J_A e K_A.

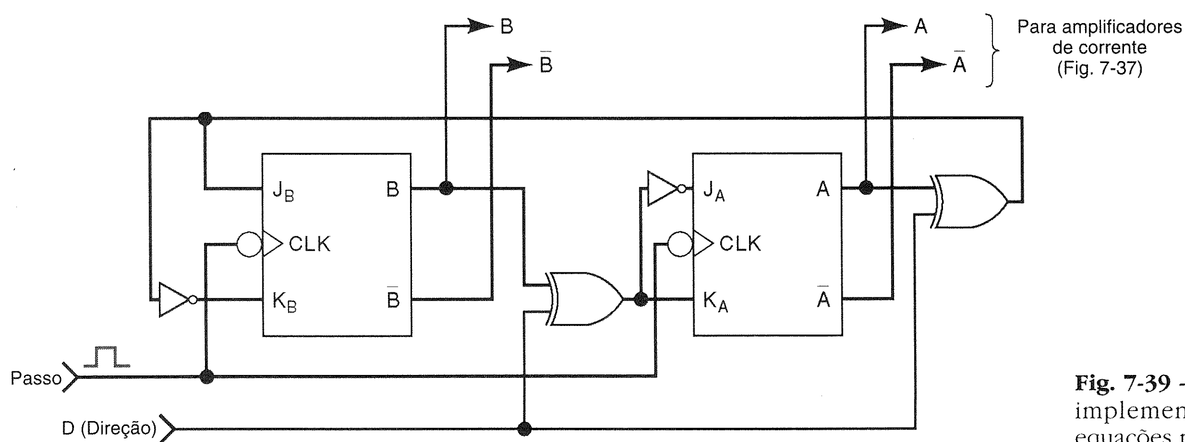


Fig. 7-39 - Contador síncrono implementado a partir das equações para *J* e *K*.

O passo final é mostrado na Fig. 7-39, onde o contador síncrono de dois bits é implementado usando as expressões para J e K obtidas nos mapas.

Questões de Revisão

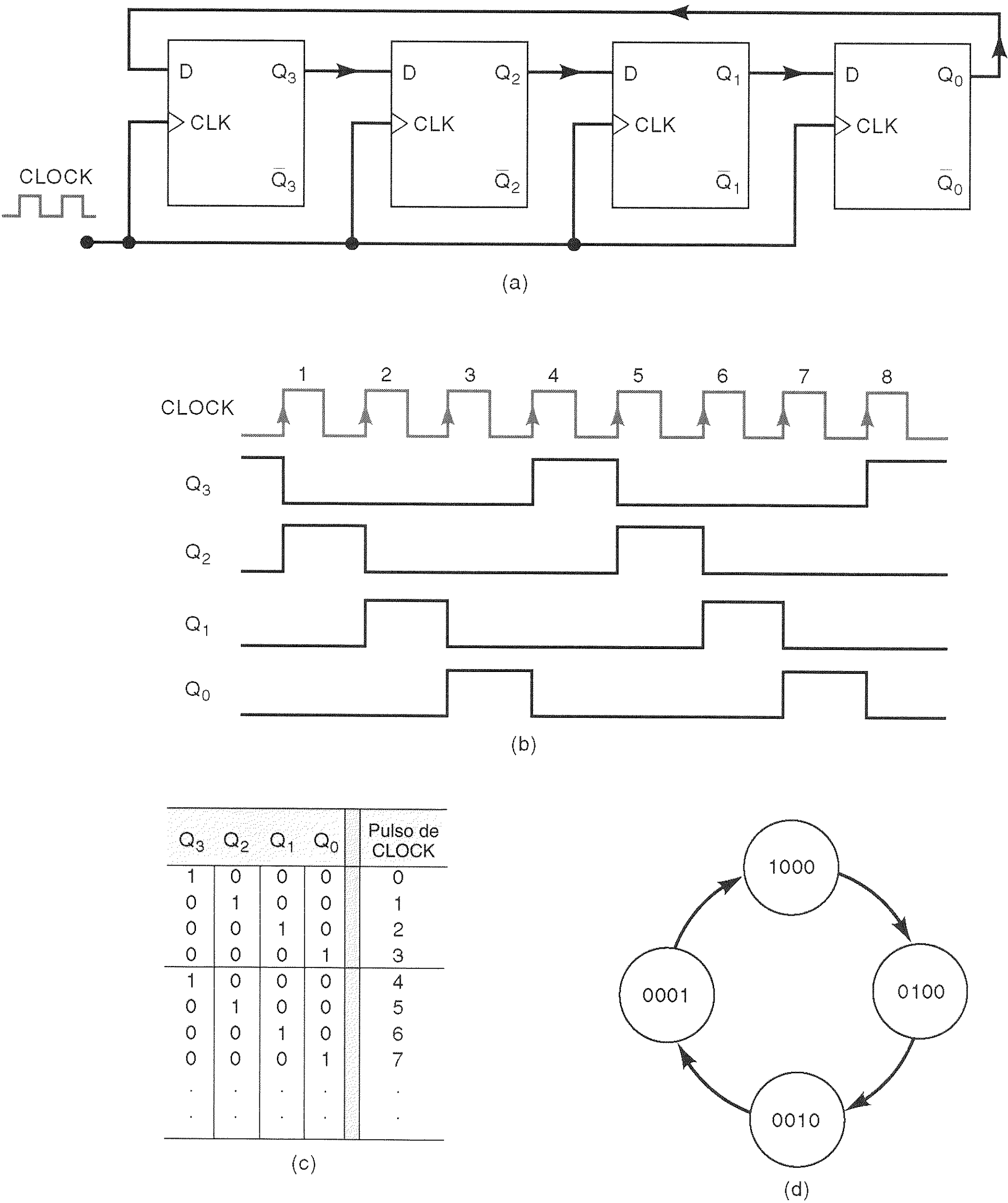
1. Enumere os seis passos no procedimento de projeto para um contador síncrono.
2. Que informação está contida na tabela de excitação J-K?
3. Que informação está contida na tabela de excitação do circuito?
4. *Verdadeiro ou falso*: O procedimento de projeto de contadores síncronos pode ser usado para a seguinte seqüência: 0010, 0011, 0100, 0111, 1010, 1110, 1111, e a partir daí repete-se o ciclo.

7-15 CONTADORES COM REGISTRADORES DE DESLOCAMENTO

Na Seção 5-18 vimos como conectar FFs para formar um registrador de deslocamento para transferir dados da esquerda para a direita, ou vice-versa, um bit de cada vez (serialmente). Contadores com registradores de deslocamento usam *realimentação*, o que significa que a saída do último FF do registrador é conectada de algum modo ao primeiro flip-flop.

Contador em Anel

O contador com registrador de deslocamento mais simples é essencialmente um **registrador de deslocamento circular** conectado de modo que o último FF desloque seu valor para o primeiro FF. Este arranjo é mostrado na Fig. 7-40 usando flip-flops do tipo D (flip-flops J-K também po-



Q_3	Q_2	Q_1	Q_0	Pulso de CLOCK
1	0	0	0	0
0	1	0	0	1
0	0	1	0	2
0	0	0	1	3
1	0	0	0	4
0	1	0	0	5
0	0	1	0	6
0	0	0	1	7
.	.	.	.	.
.	.	.	.	.

Fig. 7-40 - (a) Contador em anel de 4 bits; (b) formas de onda; (c) tabela de seqüência; (d) diagrama de estados.

dem ser usados). Os FFs são conectados de modo que a informação seja deslocada da esquerda para a direita e circule de volta de Q_0 para Q_3. Na maioria dos casos, somente um único 1 está no registrador, e este circula pelo registrador enquanto pulsos de clock forem aplicados. Por esta razão, ele é chamado **contador em anel**.

As formas de onda, a tabela de seqüência e o diagrama de estados podem ser vistos na Fig. 7-40, e mostram os diversos estados dos FFs à medida que pulsos são aplicados, considerando o estado inicial de $Q_3 = 1$ e $Q_2 = Q_1 = Q_0 = 0$. Após o primeiro pulso, o 1 foi deslocado de Q_3 para Q_2 de modo que o contador esteja em 0100. O segundo pulso produz o estado 0010, e o terceiro pulso produz o estado 0001. No *quarto* pulso de clock, o 1 é transferido de Q_0 para Q_3, resultando no estado 1000, que é, obviamente, o estado inicial. Pulsos subseqüentes farão com que a seqüência se repita.

Este contador funciona como um contador de módulo 4, uma vez que ele possui *quatro* estados distintos antes que a seqüência se repita. Apesar de este circuito não progredir segundo a seqüência de contagem binária normal, ele ainda é um contador porque cada contagem corresponde a um único conjunto de estados dos flip-flops. Observe que a saída de cada FF tem uma freqüência igual a um quarto da freqüência do clock, uma vez que ele é um contador em anel de módulo 4.

Contadores em anel podem ser construídos para qualquer módulo desejado. Um contador em anel de módulo N utiliza N flip-flops conectados segundo o arranjo da Fig. 7-40. De um modo geral, um contador em anel necessitará de mais flip-flops do que um contador binário de mesmo módulo. Por exemplo, um contador em anel de módulo 8 necessita de oito FFs, enquanto um contador binário de módulo 8 requer apenas três.

Apesar de ser menos eficiente no uso de FFs, um contador em anel ainda é útil porque ele pode ser decodificado sem o uso de portas decodificadoras. O sinal decodificado para cada estado é obtido na saída do seu flip-flop correspondente. Compare as formas de onda do contador em anel com aquelas decodificadas que podem ser vistas na Fig. 7-27. Em alguns casos, um contador em anel pode ser uma escolha melhor do que um contador binário com suas portas decodificadoras associadas. Isto é especialmente verdadeiro em aplicações onde o contador é usado para controlar a seqüência de operações em um sistema.

Partida de um Contador em Anel

Para funcionar corretamente, o contador em anel deve partir com apenas um FF no estado 1 e todos os outros no estado 0. Uma vez que os estados iniciais dos FFs são imprevisíveis quando a alimentação do circuito é ligada, o contador deve ser colocado no estado inicial desejado antes da aplicação de pulsos de clock. Um modo de fazer isto é aplicar um pulso momentâneo na entrada assíncrona $\overline{PRE}$ (por exemplo, Q_3 na Fig. 7-40) e na entrada $\overline{CLR}$ de todos os outros FFs. Um outro método é mostrado na Fig. 7-41. Quando a alimentação do circuito for ligada, o capacitor será lentamente carregado em direção a $+V_{CC}$. A saída do INVERSOR Schmitt-trigger 1 permanecerá em ALTO, e a saída do INVERSOR 2 permanecerá em BAIXO até que a tensão do capacitor exceda a tensão de limiar positivo (V_{T+}) da entrada do INVERSOR 1 (em torno de 1,7 V). Isto fará com que a entrada $\overline{PRE}$ de Q_3 e a entrada $\overline{CLR}$ de Q_2, Q_1 e Q_0 sejam mantidas em BAIXO por um tempo suficientemente grande enquanto a alimentação está sendo ligada para assegurar que o contador comece de 1000.

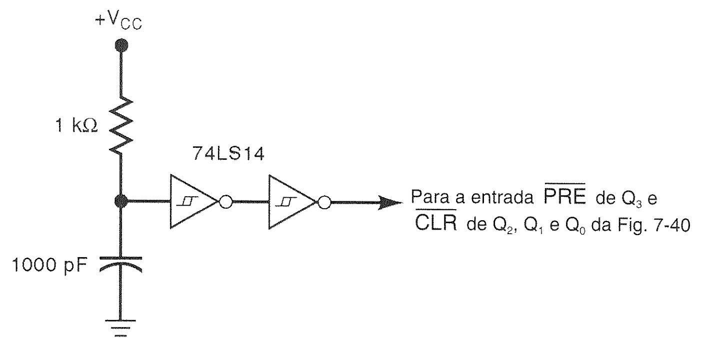


Fig. 7-41 - Circuito que assegura que o contador em anel da Fig. 7-40 inicie no estado 1000 quando a alimentação for ligada.

Contador Johnson

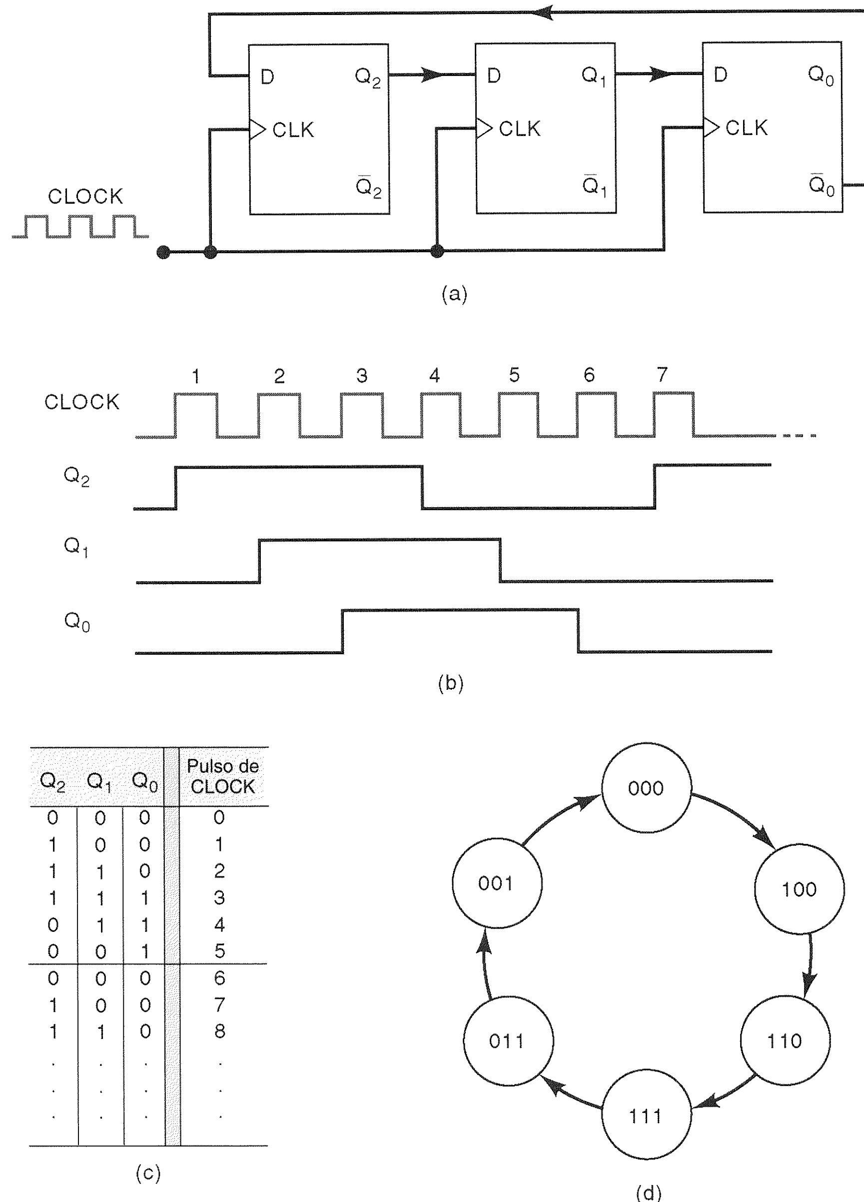
O contador em anel básico pode ser ligeiramente modificado para produzir um outro tipo de contador com registrador de deslocamento, que terá propriedades um pouco diferentes. O **contador Johnson** ou **contador em anel torcido** é construído exatamente como um contador em anel normal, exceto pelo fato de que a saída *invertida* do último FF é que está conectada à entrada do primeiro. Um contador Johnson de três bits é mostrado na Fig. 7-42. Observe que a saída $\overline{Q}_0$ é conectada de volta à entrada D de Q_2. Isto significa que o *inverso* do nível armazenado em Q_0 será transferido para Q_2 no pulso de clock.

A operação de um contador Johnson é fácil de analisar se notarmos que em cada transição positiva do pulso de clock o nível de Q_2 é deslocado para Q_1, o de Q_1 é deslocado para Q_0, e o *inverso* do nível de Q_0 é deslocado para Q_2. Usando estas idéias e considerando que todos os FFs estão inicialmente em 0, as formas de onda, a tabela de seqüência e o diagrama de estados vistos na Fig. 7-42 podem ser obtidos.

O exame das formas de onda e da tabela de seqüência revela os seguintes pontos importantes:

1. Este contador possui seis estados distintos: 000, 100, 110, 111, 011 e 001, antes que a seqüência se repita. Portanto, ele é contador Johnson de módulo 6. Observe que ele não conta conforme a contagem binária normal.
2. A forma de onda da saída de cada FF é uma onda quadrada (50% de taxa de ciclo) com um sexto da freqüência do clock. Além disso, as formas de onda de duas saídas sucessivas estão deslocadas de um período de clock.

O módulo de um contador Johnson será sempre igual a *duas* vezes o número de FFs. Por exemplo, se conectarmos cinco FFs conforme o arranjo da Fig. 7-42, teremos um contador de módulo 10, onde a saída de cada FF é uma onda quadrada com um décimo da freqüência do clock. Portanto, é possível construir um contador de módulo N (onde N é um número par) conectando $N/2$ flip-flops neste arranjo de contador.



Q_2	Q_1	Q_0	Pulso de CLOCK
0	0	0	0
1	0	0	1
1	1	0	2
1	1	1	3
0	1	1	4
0	0	1	5
0	0	0	6
1	0	0	7
1	1	0	8
.	.	.	.
.	.	.	.
.	.	.	.

(c)

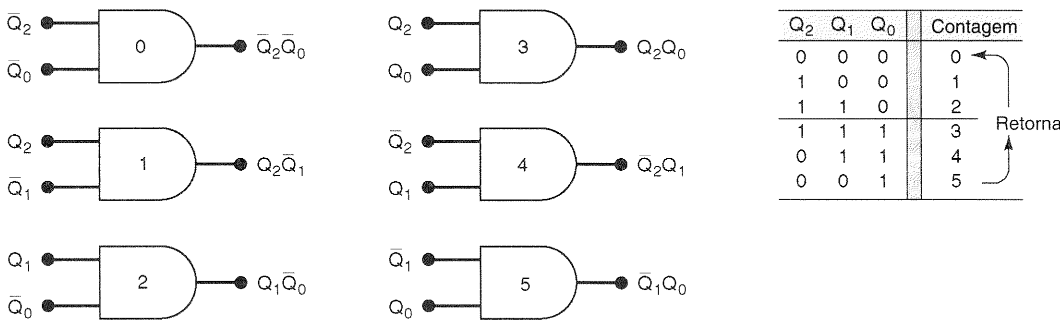
Fig. 7-42 - (a) Contador Johnson de módulo 6; (b) formas de onda; (c) tabela de seqüência; (d) diagrama de estado.

Decodificando um Contador Johnson

Para um dado módulo, um contador Johnson necessita de apenas metade do número de FFs que um contador em anel. Entretanto, um contador Johnson necessita de portas decodificadoras, enquanto um contador em anel não. Como um contador binário, o contador Johnson usa uma porta lógica para decodificar cada contagem, mas necessita apenas de portas de duas entradas, independentemente do número de flip-flops que existam. A Fig. 7-43 mostra as portas decodificadoras para os seis estados do contador Johnson da Fig. 7-42.

Observe que cada porta decodificadora possui apenas duas entradas, mesmo havendo três FFs no contador. Isto acontece porque, para cada contagem, dois dos três FFs estão em uma combinação única de estados. Por exemplo, a combinação $Q_2 = Q_0 = 0$ ocorre apenas uma vez na seqüência, na contagem 0. Então, a porta AND 0, com entradas $\overline{Q}_2$ e $\overline{Q}_0$, pode ser usada para decodificar esta contagem. Esta mesma característica é compartilhada por todos os outros estados da seqüência, como o leitor pode verificar. De fato, *qualquer* que seja o tamanho do contador Johnson, as portas decodificadoras terão apenas duas entradas.

Contadores Johnson representam um meio-termo entre contadores binários e contadores em anel. Um contador Johnson requer um menor número de FFs que um contador em anel; entretanto, geralmente necessita de um maior número que um contador binário. Ele possui mais circuitos decodificadores do que um contador em anel, porém menos do que um contador binário. Portanto, ele às vezes representa uma escolha lógica para certas aplicações.



Q_2	Q_1	Q_0	Contagem
0	0	0	0
1	0	0	1
1	1	0	2
1	1	1	3
0	1	1	4
0	0	1	5

Fig. 7-43 - Lógica de decodificação para um contador Johnson de módulo 6.

CIs Contadores com Registradores de Deslocamento

Existem muito poucos contadores em anel ou contadores Johnson disponíveis como circuitos integrados. A razão para isto é que é relativamente simples pegar um CI registrador de deslocamento e conectá-lo ou como um contador em anel ou como um contador Johnson (veja o Exemplo 7-20). Alguns dos CIs contadores Johnson CMOS (74HC4017, 74HC4022) incluem o circuito de decodificação completo no mesmo chip do contador.

Questões de Revisão

1. Que contador com registrador de deslocamento necessita de um maior número de FFs para um dado módulo?
2. Que contador com registrador de deslocamento necessita de mais circuitos decodificadores?
3. Como um contador em anel pode ser convertido para um contador Johnson?
4. *Verdadeiro ou falso*:
   (a) As saídas de um contador em anel são sempre ondas quadradas.
   (b) O circuito decodificador para um contador Johnson é mais simples do que para um contador binário.
   (c) Contadores em anel e Johnson são contadores síncronos.

RESUMO (PARTE I)

1. Em contadores assíncronos, o sinal de clock é aplicado ao FF LSB, e todos os outros FFs são disparados pela saída do flip-flop anterior.
2. O módulo de um contador é o número de estados estáveis durante o seu ciclo de contagem. Ele também é a maior razão de divisão de freqüência.
3. O módulo normal de um contador é 2^N. Uma maneira de modificar o módulo do contador é adicionar circuitos que façam com que ele recicle antes de atingir a sua última contagem normal.
4. Contadores podem ser ligados em cascata para produzir intervalos de contagem maiores e uma maior razão de divisão de freqüência.
5. Em um contador síncrono, todos os FFs são disparados simultaneamente pelo mesmo sinal de clock.
6. A freqüência máxima do clock para um contador assíncrono, f_{max}, decresce à medida que o número de bits aumenta. Para um contador síncrono, f_{max} permanece a mesma, independentemente do número de bits.
7. Um contador decádico é qualquer contador de módulo 10. Um contador BCD é um contador decádico cuja seqüência contém os dez códigos BCD (0-9).
8. Um contador com carga pode ser carregado com qualquer valor de contagem desejado para a partida.
9. Um contador crescente/decrescente pode ser comandado pela entrada de contagem crescente ou pela entrada de contagem decrescente.
10. Portas lógicas podem ser usadas para decodificar (detectar) algum ou todos os estados de um contador.
11. Contadores assíncronos podem produzir glitches nas portas decodificadoras, devido aos atrasos de propagação do contador. Contadores síncronos são menos propensos a causar glitches de decodificação. Amostragem é uma técnica para eliminar os efeitos dos glitches de decodificação.
12. Contadores síncronos com seqüência de contagem arbitrária podem ser implementados usando um procedimento de projeto padrão.
13. Um contador em anel é, na verdade, um registrador de deslocamento de N bits que recircula continuamente um único 1 e portanto age como um contador de módulo N. Um contador Johnson é um contador em anel modificado que opera como um contador de módulo $2N$.

TERMOS IMPORTANTES (PARTE I)

contador assíncrono (por pulsação)
módulo
contador decádico
contador BCD
contador crescente
contador decrescente
contadores síncronos (paralelos)
contadores crescentes/decrescentes
contadores carregáveis
carga paralela
contador de vários estágios
decodificação
glitches de decodificação
strobing (amostragem)
ligação em cascata
projeto de circuito(s) seqüencial(is)
tabela de excitação J-K
tabela de excitação do circuito
registrador de deslocamento circular
contador em anel
contador Johnson

PARTE II

7-16 APLICAÇÕES DE CONTADORES: FREQÜENCÍMETRO

Existem numerosas aplicações para os diversos tipos de contadores que temos estudado. Nesta seção e na próxima, analisaremos duas aplicações representativas que ilustram a utilização de contadores em sistemas digitais.

Um **freqüencímetro** é um circuito que pode medir e mostrar a freqüência de um sinal. Um dos métodos mais diretos para construir um freqüencímetro é mostrado, de maneira simplificada, na Fig. 7-44(a). Ela contém um contador com circuitos do tipo decodificador/display e uma porta AND. As entradas da porta AND incluem os pulsos da freqüência desconhecida, f_x, e um pulso de SAMPLE que controla por quanto tempo esses pulsos podem passar pela porta AND para o contador. O contador usualmente é feito com contadores BCD em cascata (Fig. 7-32), e a unidade decodificador/display converte as saídas BCD para a apresentação decimal a fim de facilitar a visualização.

As formas de onda na Fig. 7-44(b) mostram que um pulso de CLEAR é aplicado no contador em t_0 para iniciar o contador em 0. Antes de t_1, a forma de onda do pulso SAMPLE está em BAIXO, e portanto também a saída da porta AND, Z, está em BAIXO e o contador não está contando. O pulso SAMPLE vai para ALTO de t_1 até t_2; isto é denominado **intervalo de amostragem**. Durante o intervalo de amostragem, os pulsos da freqüência desconhecida passarão pela porta AND e serão contados pelo contador. Após t_2, a saída da AND retorna para BAIXO e o contador pára de contar. Deste modo, o contador terá contado o número de pulsos que ocorreram durante o intervalo de amostragem, e seu conteúdo resultante é uma medida direta da freqüência da forma de onda pulsada.

EXEMPLO 7-18

A freqüência desconhecida é de 3.792 pulsos por segundo (pps). O contador é levado para o estado 0 antes de t_1. Determine a saída do contador após um intervalo de amostragem de **(a)** 1 s, **(b)** 0,1 s e **(c)** 10 ms.

Solução

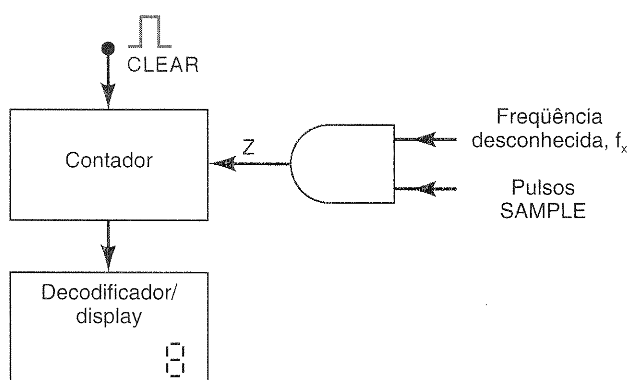
(a) Dentro de um intervalo de amostragem de 1 s existirão 3.792 pulsos entrando no contador, e portanto, depois de t_2, o conteúdo do contador será 3.792.

(b) Com um intervalo de amostragem de 0,1 s, o número de pulsos que passa através da porta AND para o contador será de 3.792 pulsos/s × 0,1 s = 379,2. Isto significa que ou 379 ou 380 pulsos serão contados, dependendo em que parte do ciclo do pulso t_1 ocorre.

(c) Com um intervalo de amostragem de 10 ms = 0,01 s, o contador apresentará uma contagem de 37 ou 38.

A exatidão deste método depende quase que inteiramente da duração do intervalo de amostragem, que deve ser controlado de modo bem preciso. Um método comumente utilizado para obter pulsos de amostragem bem precisos é mostrado na Fig. 7-45. Um oscilador controlado a cristal é usado para gerar uma forma de onda de 100 kHz bastante precisa, que é transformada em pulsos quadrados e levada para uma série de contadores decádicos, que estão sendo usados para dividir sucessivamente esta freqüência de 100 kHz por 10. As freqüências nas saídas de cada contador decádico são tão precisas (percentualmente) quanto a freqüência do cristal. Estes contadores decádicos são usualmente contadores binários ou Johnson.

A chave rotativa é usada para selecionar uma das freqüências de saída dos contadores decádicos, que por sua vez será



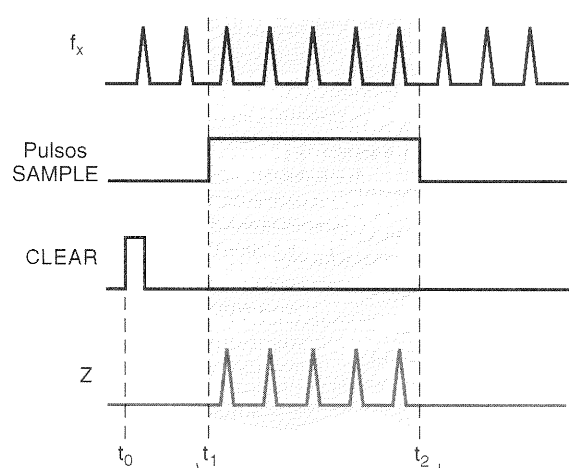


Fig. 7-44 - Método básico para contagem de freqüência.

dividida por 2 por um FF. Por exemplo, na posição 1 da chave, os pulsos de 1 Hz são levados ao flip-flop *Q*, que está operando no modo de comutação, de maneira que sua saída será uma onda quadrada com um período de $T = 2$ s e com um pulso de duração $t_p = T/2 = 1$ s. A duração deste pulso é o intervalo de amostragem desejado de 1 s. Na posição 2, o intervalo de amostragem seria 0,1 s, e assim por diante para as outras posições.

EXEMPLO 7-19

Admita que o contador na Fig. 7-44 é construído com três contadores BCD em cascata, com os respectivos displays. Se a freqüência desconhecida de entrada estiver entre 1 kpps e 9,99 kpps, qual é a melhor escolha para a posição da chave na Fig. 7-45?

Solução

Com três contadores BCD, a capacidade total do contador é 999. Uma freqüência de 9,99 kpps produziria uma contagem de 999 se um intervalo de amostragem de 0,1 s fosse usado. Assim, de modo a usar a capacidade total do contador, a chave deveria ser colocada na posição 2. Se um intervalo de amostragem de 1 s fosse usado, a capacidade do contador sempre seria excedida para as freqüências na faixa especificada. Se um intervalo de amostragem mais curto fosse utilizado, o contador contaria apenas entre 1 e 99; isto daria uma leitura com apenas dois dígitos significativos e seria um desperdício da capacidade do contador.

Freqüencímetro Completo

Vamos analisar agora um circuito de freqüencímetro mais completo na Fig. 7-46(a). O circuito contém um monoestável, um flip-flop J-K operando em comutação, e a porta AND tem três entradas, uma das quais é a saída *X* do FF. Os pulsos SAMPLE estão conectados na porta AND e também na entrada *CLK* do FF. Estes pulsos SAMPLE seriam gerados por um circuito semelhante ao da Fig. 7-45. A seguinte descrição, passo a passo, se refere às formas de onda na Fig. 7-46(b).

1. Admita que o flip-flop *X* está no estado 0 (ele comutou para 0 na descida do pulso de amostragem anterior).
2. Este nível BAIXO de *X* é levado para a porta AND, desabilitando sua saída, de modo que nenhum pulso é levado para o contador, mesmo quando o primeiro pulso SAMPLE ocorre entre t_1 e t_2.
3. Em t_2, a descida do primeiro pulso de SAMPLE comuta o flip-flop *X* para o estado 1 (note que $J = K = 1$). Esta transição positiva em *X* dispara o MONO, que gera um pulso de 100 ns para *limpar* o contador. O contador agora apresenta *zero*.
4. Em t_3, o segundo pulso de SAMPLE habilita a porta AND (já que *X* agora está em 1) e permite que a freqüência desconhecida alcance o contador para ser contada até t_4.
5. Em t_4, o pulso de SAMPLE retorna para o nível BAIXO e comuta *X* para BAIXO, desabilitando a porta AND. O contador pára de contar.
6. Entre t_4 e t_6, o contador mantém e mostra a contagem que alcançou até t_4. Note que o terceiro pulso de SAMPLE não habilita a porta AND, pois o flip-flop *X* está em BAIXO.
7. Em t_6, a descida do pulso SAMPLE comuta *X* para ALTO, e a operação segue a mesma seqüência que começou em t_2.

Esse freqüencímetro, portanto, realiza uma seqüência repetitiva de limpar para 0, contar, manter o valor para o display, limpar para 0, contar e assim por diante. Por exemplo, vamos supor que o contador tenha três estágios BCD e um display de três dígitos. Se usarmos um intervalo de amostragem de 1 s e a freqüência desconhecida for de 237

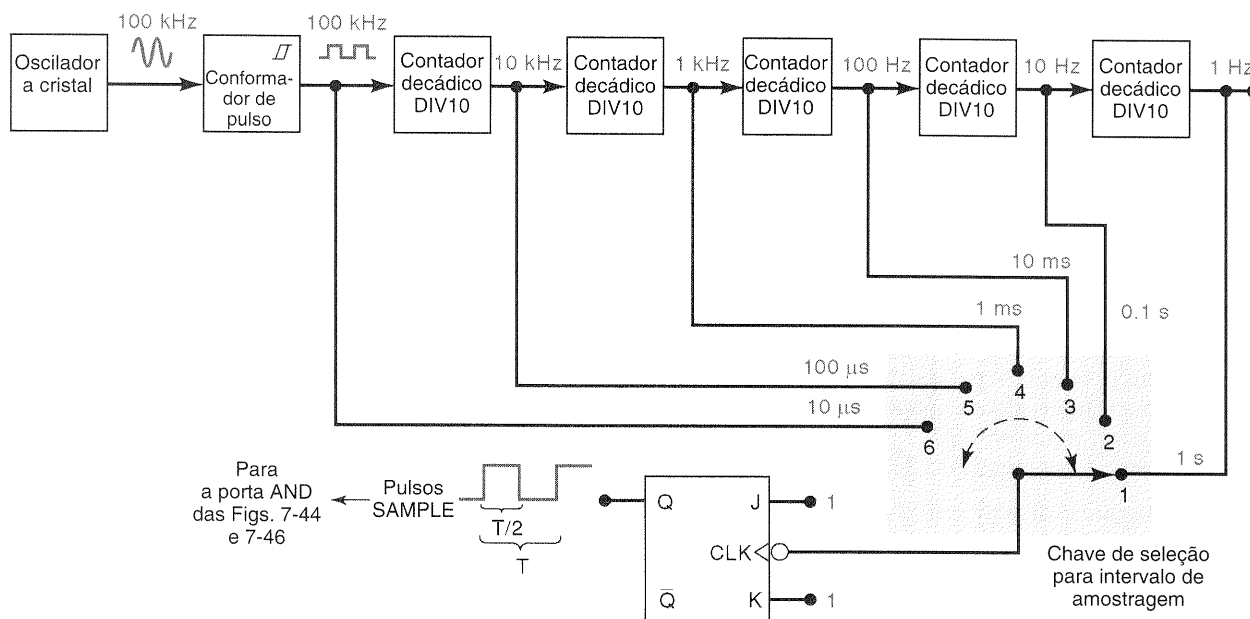


Fig. 7-45 - Método para obter intervalos de amostragem precisos para um freqüencímetro.

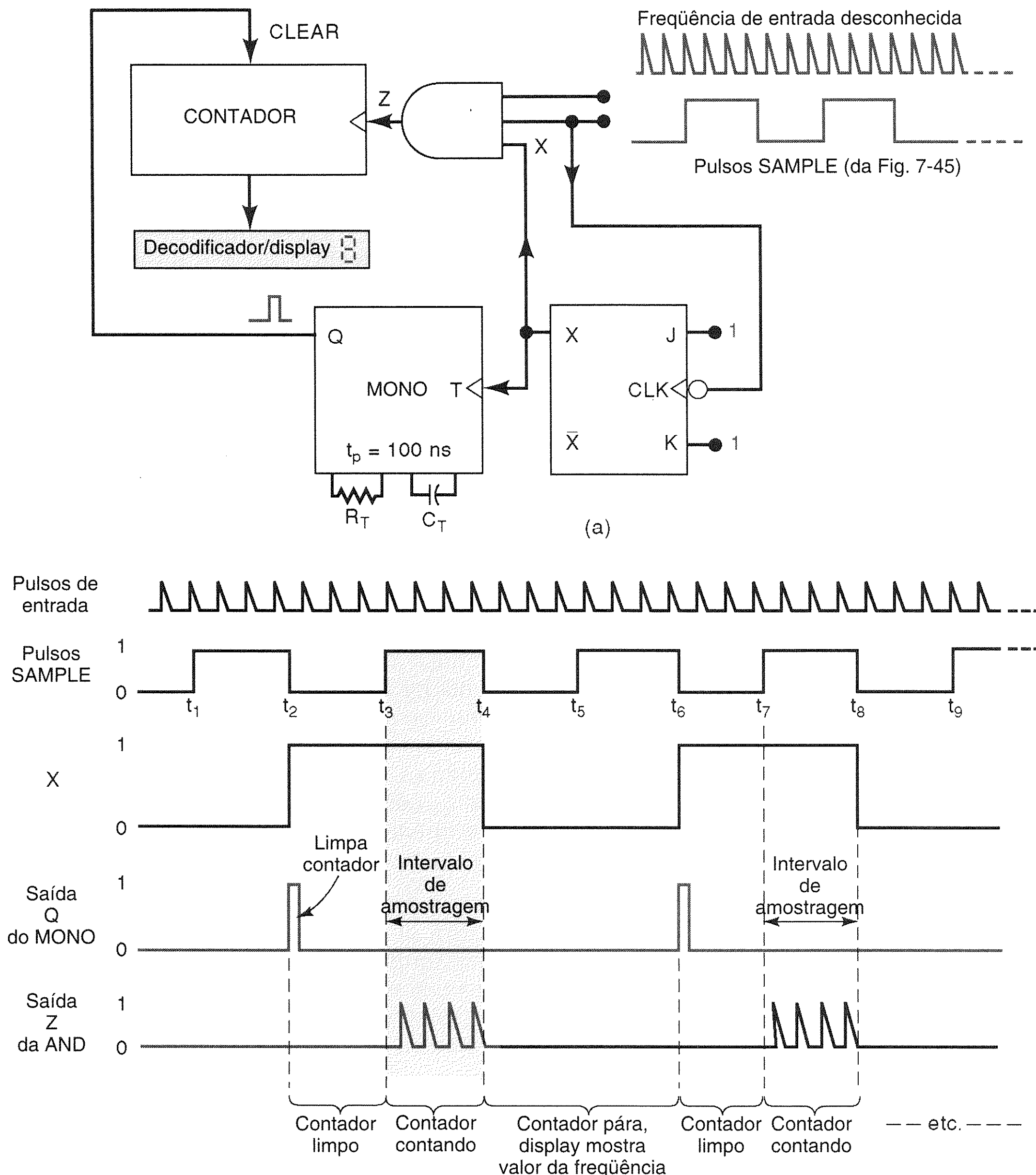


Fig. 7-46 - Freqüencímetro.

pps, o contador e o display seguirão indefinidamente esta seqüência:

- Limpa para 0 e mostra 0 por 1 s [t_2 até t_3 na Fig. 7-46(b)].
- Começando em 0, conta os pulsos da freqüência desconhecida durante o intervalo de amostragem de 1 s (t_3 até t_4); a contagem vai parar em 237.
- Mantém e mostra a contagem de 237 por 2 s (t_4 até t_6).

Tendo em vista que o display está conectado diretamente nas saídas do contador, o display mostrará a operação de reset e a contagem. Isto torna muito difícil ler o display para determinar a freqüência desconhecida, exceto para intervalos de amostragem muito lentos. Este problema pode ser resolvido inserindo-se um *registrador buffer* entre o contador e a unidade do decodificador/display. Consideraremos esta característica no Problema 7-43.

QUESTÕES DE REVISÃO

1. Qual é o melhor intervalo de amostragem para usar se o contador de pulsos tiver quatro estágios BCD e a freqüência de entrada estiver entre 2 e 8 Mpps?
2. Descreva a seqüência de operações do freqüencímetro completo da Fig. 7-46.
3. Por que não seria inteligente utilizar contadores em anel, em vez de contadores Johnson, para os contadores decádicos na Fig. 7-45?

7-17 APLICAÇÕES DE CONTADORES: RELÓGIO DIGITAL

Uma das aplicações mais comuns de contadores é o relógio digital — um relógio de tempo que mostra em um display o tempo do dia em horas, minutos e às vezes segundos. De modo a construir um relógio digital preciso, é necessária uma freqüência básica bastante controlada. Para relógios digitais que funcionam com bateria, a freqüência básica é obtida normalmente de um oscilador com cristal de quartzo. Relógios digitais que operam com tensão AC, da rede de energia elétrica, podem usar a freqüência de 60 Hz como sua freqüência básica. Em ambos os casos, a freqüência básica deve ser dividida para a freqüência de 1 Hz ou 1 pulso por segundo (1 pps). A Fig. 7-47 mostra o diagrama de blocos básico para um relógio digital que opera com 60 Hz.

O sinal de 60 Hz é enviado para um circuito Schmitt-trigger para produzir pulsos quadrados a uma taxa de 60 pps. Esta forma de onda de 60 pps é levada ao contador de módulo 60, que é usado para dividir os 60 pps para 1 pps. O sinal de 1 pps é levado para a seção de SEGUNDOS, que por sua vez é usada para contar e mostrar os segundos de 0 até 59. O contador BCD avança uma contagem por segundo. Após 9 segundos, o contador BCD recicla para 0, o que, por sua vez, aciona o contador de módulo 6 e faz com que ele avance uma contagem. Isto continua por 59 segundos; neste ponto, o contador de módulo 6 está com a contagem de 101 (5) e o contador BCD está com 1001 (9), portanto o display apresenta 59 segundos. O próximo pulso recicla o contador BCD para 0, o que por sua vez recicla o contador de módulo 6 para 0 (lembre-se, o contador de módulo 6 conta de 0 a 5).

A saída do contador de módulo 6 da seção dos SEGUNDOS tem uma freqüência de 1 pulso por minuto (ele recicla a cada 60 segundos). Este sinal é levado para a seção dos MINUTOS, que conta e mostra os minutos de 0 até 59. A seção dos MINUTOS é idêntica à seção dos segundos, e opera exatamente da mesma maneira.

A saída do contador de módulo 6 da seção dos MINUTOS tem uma freqüência de 1 pulso por hora (ele recicla a cada 60 minutos). Este sinal é levado para a seção das HORAS, que conta e mostra as horas de 1 até 12. A seção das HORAS é diferente das seções dos SEGUNDOS e dos MINUTOS, pois ela nunca vai para o estado 0. Os circuitos desta seção são suficientemente não-usuais para garantirem uma investigação mais detalhada.

A Fig. 7-48 mostra em detalhe os circuitos contidos na seção das HORAS. Ela tem um contador BCD para contar as unidades das horas, e um único FF (módulo 2) para contar dezenas de horas. O contador BCD é um 74LS192, que opera exatamente igual ao 74LS193, que estudamos anteriormente, com a exceção de que ele só conta entre 0000 e 1001. Em outras palavras, o 74LS192 pode tanto contar em BCD de modo crescente (isto é, de 0 a 9, reciclando para 0) como de modo decrescente (isto é, de 9 para 0, reciclando para 9). Aqui, ele é usado para contar no modo crescente, em resposta ao sinal de 1 pulso/hora vindo da seção de MINUTOS. O INVERSOR na entrada CP_U é necessário pois o 74LS192 responde a transições de subida, e queremos que ele responda na descida, que ocorre quando a seção de MINUTOS recicla de volta para 0.

Os pulsos que chegam vão avançando a contagem BCD uma vez por hora. Por exemplo, às 7 horas este contador estará com 0111, e seu circuito decodificador/display estará mostrando o número 7. Ao mesmo tempo, X estará em BAIXO, e seu display mostrará um 0. Assim, os dois displays mostrarão "07". Quando o contador BCD está no estado 1001 (9) e o próximo pulso de entrada ocorre, ele reciclará de volta para 0000. A descida de Q_3 comutará o flip-flop X de 0 para 1. Isto produz o número 1 no display de X, e o número 0 no display do contador BCD, de modo que os displays combinados apresentam "10", para indicar 10 horas em ponto.

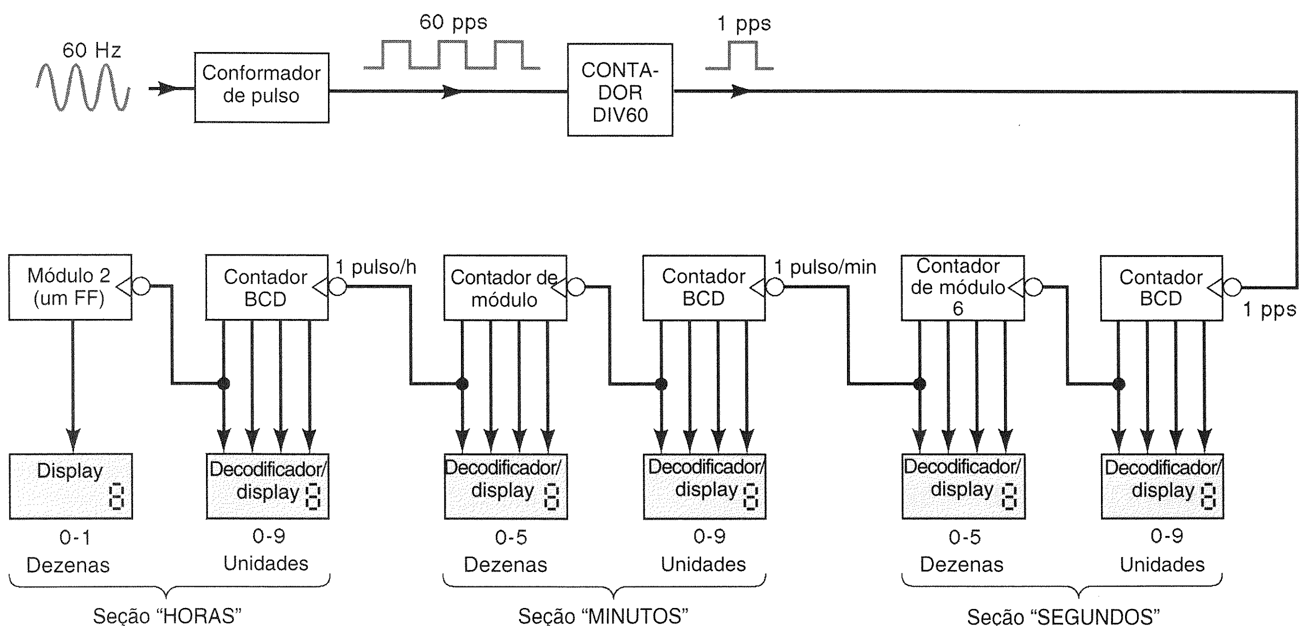


Fig. 7-47 - Diagrama de blocos para um relógio digital.

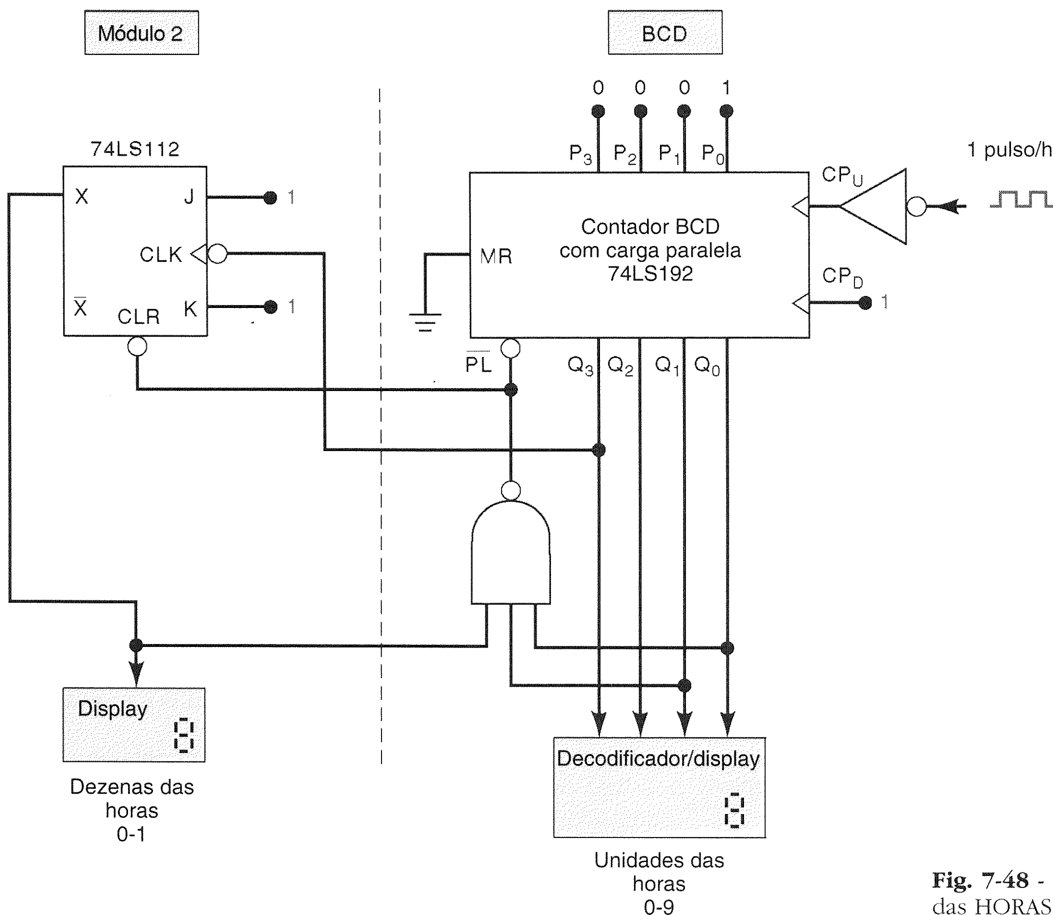


Fig. 7-48 - Circuito detalhado para a seção das HORAS.

Os dois próximos pulsos incrementam o contador BCD, de modo que "11" e "12" são apresentados às 11 e às 12 horas, respectivamente. O próximo pulso leva o contador BCD para 0011 (3). Neste estado, as saídas Q_1 e Q_0 do contador estão ambas em ALTO, e X ainda está em ALTO. Assim, a saída da porta NAND vai para BAIXO e ativa o $\overline{CLR}$ do flip-flop X e a entrada $\overline{PL}$ do 74LS192. Isto limpa X para 0 e carrega o contador BCD com 0001. O resultado é a apresentação no display de "01", para indicar 1 hora. Diversos problemas no final do capítulo fornecerão maiores detalhes sobre o circuito do relógio digital.

QUESTÕES DE REVISÃO

1. Relacione os blocos básicos que formam um circuito de relógio digital.
2. Por que é necessário um INVERSOR na Fig. 7-48?

7-18 CIRCUITOS INTEGRADOS DE REGISTRADORES

Os vários tipos de registradores podem ser classificados de acordo com a maneira pela qual os dados podem ser apresentados ao registrador para armazenamento e com a maneira pela qual os dados saem do registrador. As diversas classificações são listadas a seguir:

1. Entrada paralela/saída paralela
2. Entrada serial/saída serial
3. Entrada paralela/saída serial
4. Entrada serial/saída paralela

Cada um destes tipos e muitas variações estão disponíveis sob a forma de CIs, de modo que o projetista pode, na maioria das vezes, encontrar exatamente o que é necessário para uma dada aplicação. Nas seções seguintes, examinaremos um CI representativo para cada uma das categorias relacionadas.

7-19 ENTRADA PARALELA/SAÍDA PARALELA — O 74174 E O 74178

Realmente existem dois tipos de registradores na categoria de **entrada paralela/saída paralela**. Um é estritamente paralelo, e o outro é, na verdade, um registrador de deslocamento que pode ser carregado com dados paralelos e tem saídas paralelas disponíveis.

O 74174

A Fig. 7-49(a) mostra o diagrama lógico para o 74174 (também para o 74LS174 e o 74HC174), um registrador de 6 bits

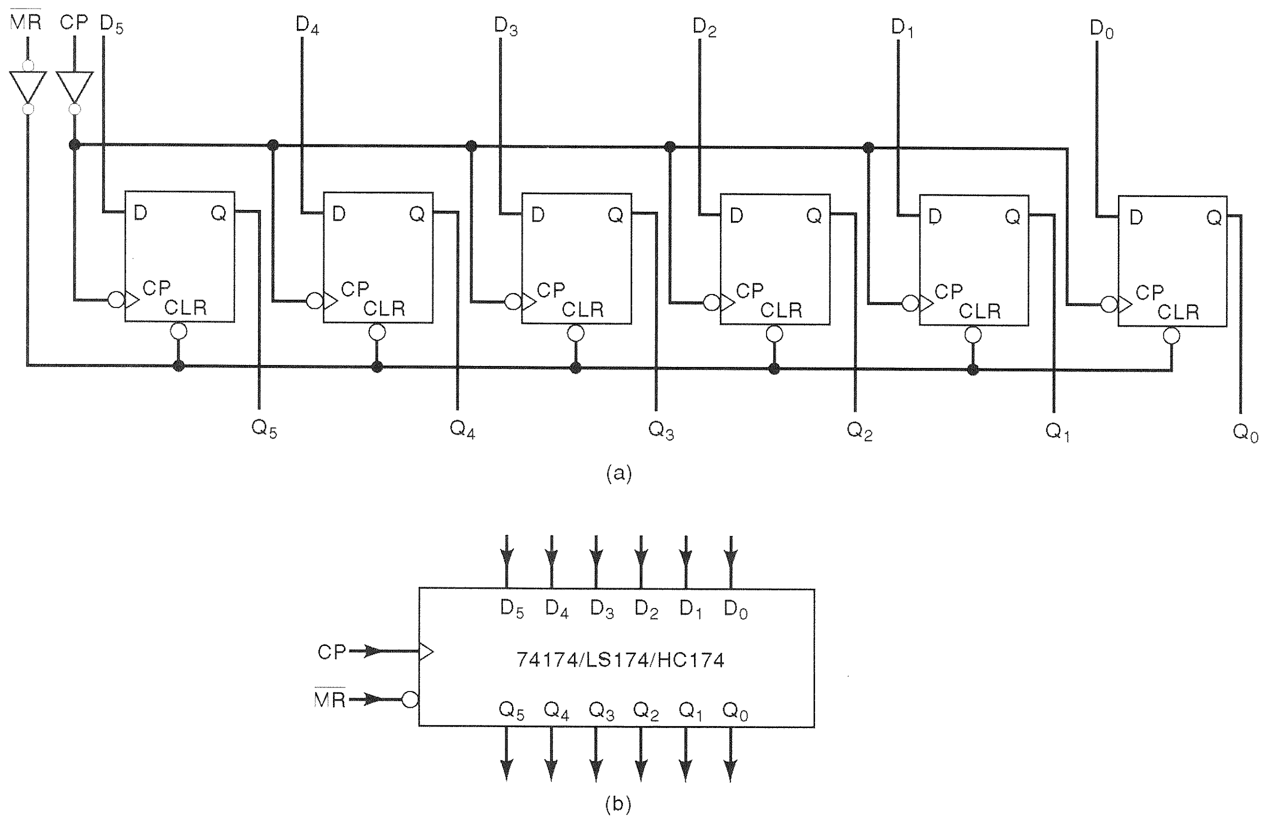


Fig. 7-49 - (a) Diagrama do circuito do 74174; (b) símbolo lógico. (Cortesia da Fairchild, uma companhia da Schlumberger.)

que tem entradas paralelas, D_5 até D_0, e saídas paralelas, Q_5 até Q_0. Os dados paralelos são carregados no registrador na subida da entrada de clock *CP*. Uma entrada de reset geral, $\overline{MR}$, pode ser usada para ressetar assincronamente todos os FFs do registrador para 0.

O símbolo lógico para o 74174 está mostrado na Fig. 7-49(b). Este símbolo é usado em diagramas de circuitos para representar o circuito da Fig. 7-49(a).

O 74178

A Fig. 7-50(a) mostra o diagrama lógico para o 74178 (também para o 74HC178), um registrador de deslocamento de 4 bits que tem entrada paralela de dados (P_0 até P_3) e saídas paralelas (Q_0 até Q_3). Ele tem uma entrada serial de dados, D_S, e duas entradas de habilitação: *PE* (habilitador paralelo) e *SE* (habilitador serial).

A Fig. 7-50(b) é a tabela de seleção de modo que descreve os vários modos de operação para este CI. A primeira linha indica as condições de entrada necessárias para a operação de deslocamento à direita. Com $SE = 1$, os dados serão deslocados da esquerda para a direita na descida da entrada de clock $\overline{CP}$, não importando o nível lógico na entrada *PE* (lembre-se de que *X* representa a condição *don't care*). Você pode verificar isto seguindo o diagrama lógico e reparando que quando $SE = 1$, a entrada D_S atravessa as portas lógicas e aparece na entrada *D* do flip-flop Q_0. Analogamente, Q_0 aparecerá na entrada *D* de Q_1; Q_1 aparecerá na entrada *D* de Q_2; e Q_2 aparecerá na entrada *D* de Q_3.

A segunda linha da tabela indica as condições necessárias para produzir uma transferência paralela das entradas de dados paralelos (P_0 até P_3) para as saídas (Q_0 até Q_3). Quando $SE = 0$ e $PE = 1$, esta transferência paralela ocorre na descida de $\overline{CP}$. Note que esta é uma transferência *síncrona*. Novamente, acompanhando o diagrama lógico, você pode verificar que quando $SE = 0$ e $PE = 1$, as entradas paralelas de dados passarão pelas portas e aparecerão nas entradas *D* de seus respectivos FFs.

A última linha da tabela indica que a condição $SE = 0$ e $PE = 0$ faz com que os FFs do registrador mantenham seus níveis atuais, não importando o que aconteça na entrada de clock. Para esta condição de entrada, cada saída de FF é habilitada a passar por suas respectivas portas lógicas e aparecer na entrada *D* do mesmo FF. Assim, uma descida em $\overline{CP}$ não mudará o estado do FF.

A Fig. 7-51 mostra o símbolo lógico para o 74178/74HC178.

EXEMPLO 7-20

Descreva como o 74178 pode ser conectado para operar como um contador em anel e como um contador Johnson.

Solução

As conexões para contador em anel estão mostradas na Fig. 7-52. A saída Q_3 é conectada de volta para a entrada D_S. As

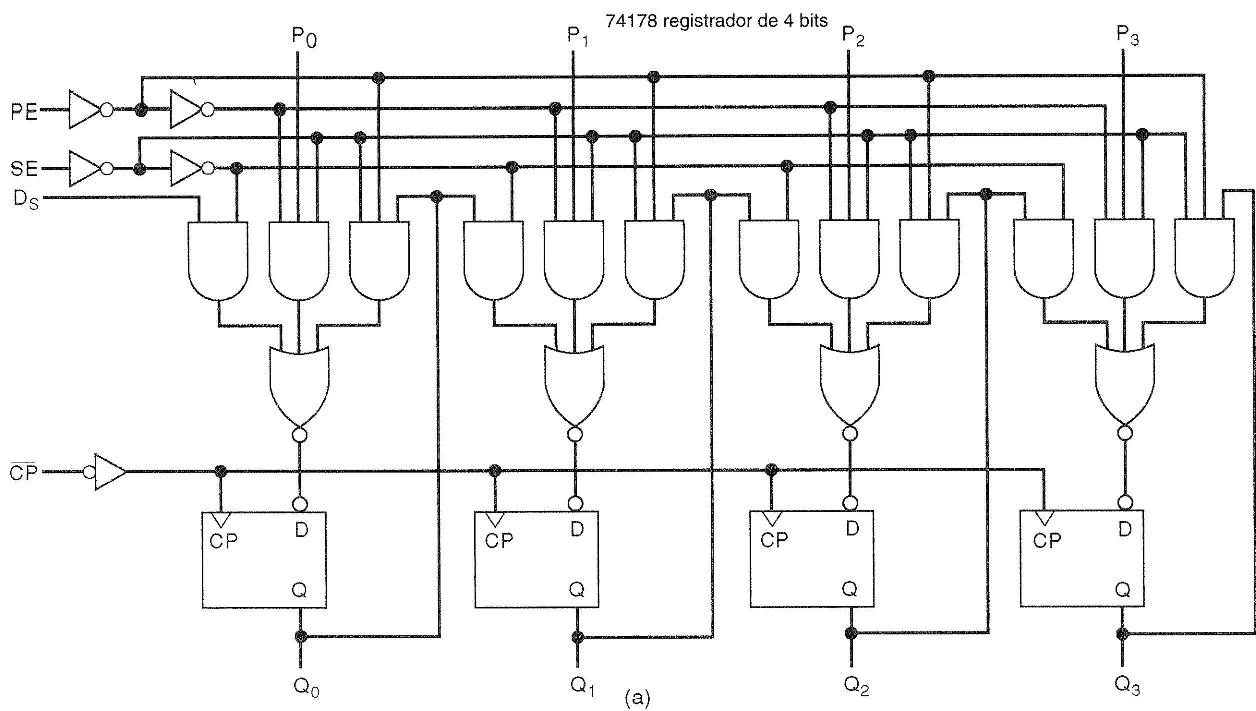


Tabela de seleção de modo

Entradas			Resposta
SE	PE	$\overline{CP}$	
H	X	⌐⌙	Deslocamento à direita $D_S \rightarrow Q_0$; $Q_0 \rightarrow Q_1$ etc.
L	H	⌐⌙	Carga paralela $P_0 \rightarrow Q_0$; $P_1 \rightarrow Q_1$ etc.
L	L	X	Mantém (Q's não mudam)

H = nível ALTO (HIGH)
L = nível BAIXO (LOW)
X = irrelevante

(b)

Fig. 7-50 - (a) Diagrama lógico para o 74178; (b) tabela de seleção de modo. (Cortesia da Fairchild, uma companhia da Schlumberger.)

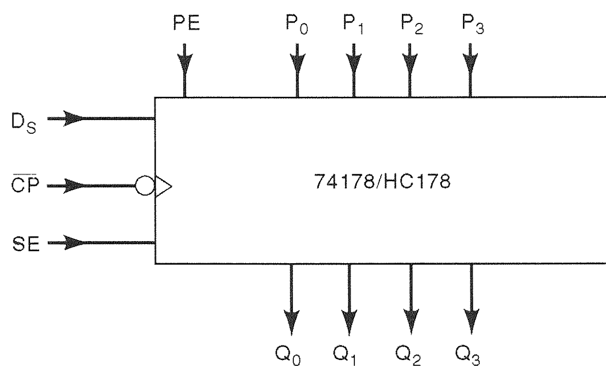


Fig. 7-51 - Símbolo lógico para o 74178/HC178.

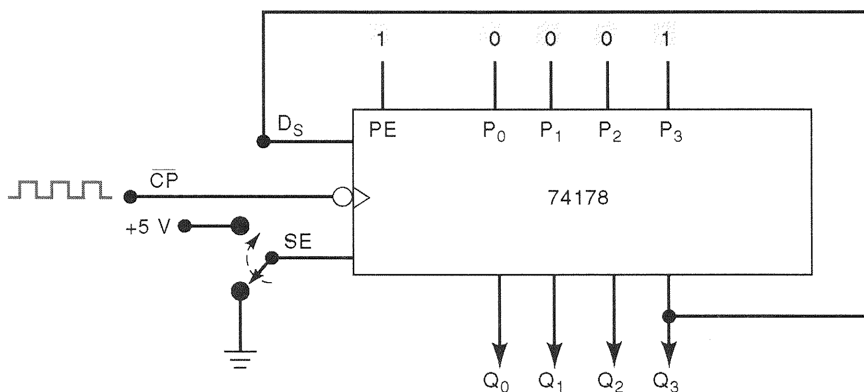


Fig. 7-52 - O 74178 ligado como um contador em anel.

entradas paralelas de dados estão permanentemente configuradas com 0001, e a entrada *PE* está conectada permanentemente em ALTO. A entrada *SE* está inicialmente em BAIXO. Isto permite que cada descida de $\overline{CP}$ faça uma carga paralela de 0001 no registrador de deslocamento. Então, *SE* é chaveado para o estado ALTO (modo de deslocamento), onde permanece, as transições de descida em $\overline{CP}$ provocam o deslocamento do único 1, que recircula pelo registrador.

Para formar um contador Johnson, adicione um INVERSOR entre a saída Q_3 e D_S e altere as entradas paralelas de dados para 0000, para ajustar o contador para 0000.

7-20 ENTRADA SERIAL/SAÍDA SERIAL — O 4731B

O 4731B é um registrador de deslocamento CMOS *quádruplo* de 64 bits da categoria **entrada serial/saída serial**. Ele contém quatro registradores de deslocamento idênticos de 64 bits em um chip. A Fig. 7-53 mostra o diagrama lógico para um dos registradores de 64 bits. Ele tem uma entrada serial, D_S, uma entrada de clock $\overline{CP}$ que responde nas descidas e uma saída serial do último FF, Q_{63}. Esta é a única saída que é acessível externamente. Note que esta saída passa por um circuito *buffer* (símbolo triangular sem a bolha de inversão). Um buffer não altera o nível lógico do sinal; ele é utilizado para fornecer uma capacidade de corrente de saída maior do que a normal. Repare também que não existe nenhuma maneira para efetuar a entrada de dados paralelos nos FFs do registrador.

EXEMPLO 7-21

Um registrador de deslocamento é utilizado freqüentemente para atrasar um sinal digital por um número inteiro de ciclos de clock. O sinal digital é aplicado na entrada serial do registrador de deslocamento e é deslocado através do registrador de deslocamento pelos sucessivos pulsos de clock até que ele alcance o final do registrador, onde ele aparece como sinal de saída. Isto está ilustrado na Fig. 7-54 usando um dos registradores de 64 bits do chip 4731B.

Vamos admitir que a entrada serial, D_S, ficou em BAIXO por um longo tempo e que pulsos de clock vinham sendo aplicados, de modo que o registrador está preenchido com 0s e Q_{63} inicia com nível BAIXO, conforme mostrado. Então, D_S vai para ALTO imediatamente antes de t_0. As descidas de $\overline{CP}$ fazem com que este nível ALTO seja deslocado através do registrador, fazendo cada FF ir para ALTO, até que finalmente em t_{63} a saída Q_{63} vai para ALTO. O efeito resultante, então, é que a mudança no sinal D_S não foi sentida na saída Q_{63} até aproximadamente 64 ciclos de clock mais tarde.

Este método para atrasar um sinal digital é comum no campo das comunicações digitais. Por exemplo, o sinal digital deve ser a versão *digitalizada* de um sinal de áudio, que deve ser atrasado antes de ser transmitido.

7-21 ENTRADA PARALELA/SAÍDA SERIAL — O 74165/74LS165/74HC165

O símbolo lógico para o 74165, um registrador de 8 bits com **entrada paralela/saída serial**, é mostrado na Fig. 7-55(a).

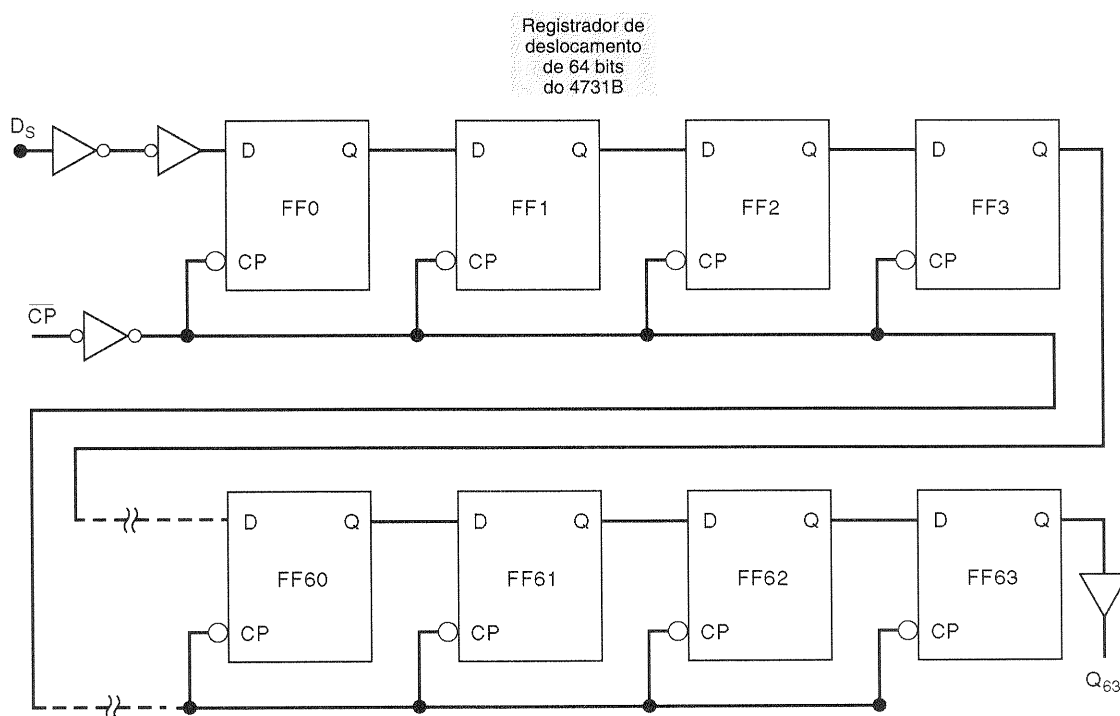


Fig. 7-53 - Diagrama lógico para um dos quatro registradores de deslocamento de 64 bits de um 4731B. (Cortesia da Fairchild, uma companhia da Schlumberger.)

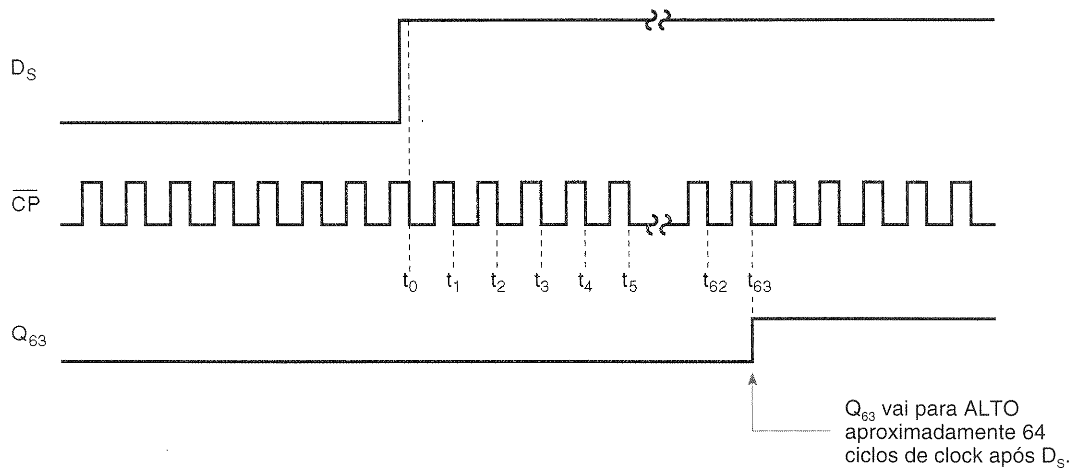


Fig. 7-54 - Formas de onda mostrando como um dos registradores de deslocamento do 4731B pode ser usado para atrasar um sinal digital.

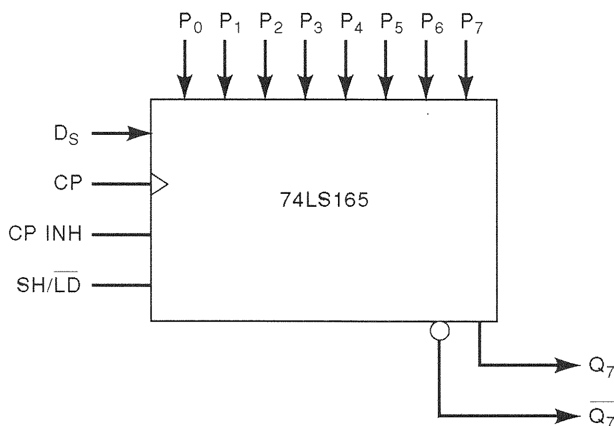


Fig. 7-55 - Símbolo lógico para o registrador de entrada paralela/saída serial 74LS165.

Ele, na verdade, tem tanto entrada de dados serial, via D_S, quanto entrada de dados paralelos, via P_0 a P_7. As únicas saídas acessíveis são Q_7 e $\overline{Q}_7$. *CP* é a entrada de clock usada para a operação de deslocamento. A entrada de inibição do clock, *CP INH*, pode ser usada para anular os efeitos dos pulsos em *CP*. A entrada de deslocamento/carga, $SH/\overline{LD}$, determina qual operação está sendo realizada.

EXEMPLO 7-22

Examine o símbolo lógico do 74LS165 e determine as condições necessárias para carregar o registrador com dados paralelos.

Solução

Já que existe uma barra de inversão sobre o símbolo $\overline{LD}$ da entrada $SH/\overline{LD}$, a operação de carga paralela é feita quando $SH/\overline{LD}$ está em BAIXO. A operação de carga é assíncrona, o que significa que acontece independentemente da entrada de clock. Os dados presentes nas entradas P_0-P_7 serão carregados nos oito FFs do registrador tão logo $SH/\overline{LD}$ vá para BAIXO. Naturalmente, apenas a saída do bit mais à direita, FF Q_7, estará externamente disponível.

EXEMPLO 7-23

Que condições são necessárias para o 74LS165 realizar uma operação de deslocamento para a direita?

Solução

A entrada de $SH/\overline{LD}$ deve estar em ALTO para a operação de deslocamento. A entrada *CP INH* deve estar em BAIXO; senão, os pulsos de clock em *CP* estariam impedidos de acionar os FFs do registrador. Quando estas condições são satisfeitas, os dados serão deslocados a cada transição de subida de cada pulso de clock, com D_S fornecendo os dados para o FF Q_0.

7-22 ENTRADA SERIAL/SAÍDA PARALELA — O 74164/74LS164/74HC164

O diagrama lógico para o 74164 é mostrado na Fig. 7-56(a). Ele é um registrador de deslocamento de oito bits com **entrada serial/saída paralela**, com a saída de cada FF externamente acessível. Em vez de uma única entrada serial, uma porta AND combina as entradas *A* e *B* para produzir a entrada serial para o flip-flop Q_0.

A operação de deslocamento ocorre nas transições de subida da entrada de clock *CP*. A entrada $\overline{MR}$, quando está em nível BAIXO, proporciona o reset assíncrono de todos os FFs.

EXEMPLO 7-24

Admita que o conteúdo inicial do registrador 74164 na Fig. 7-57(a) é 00000000. Determine a seqüência de estados conforme os pulsos de clock vão sendo aplicados.

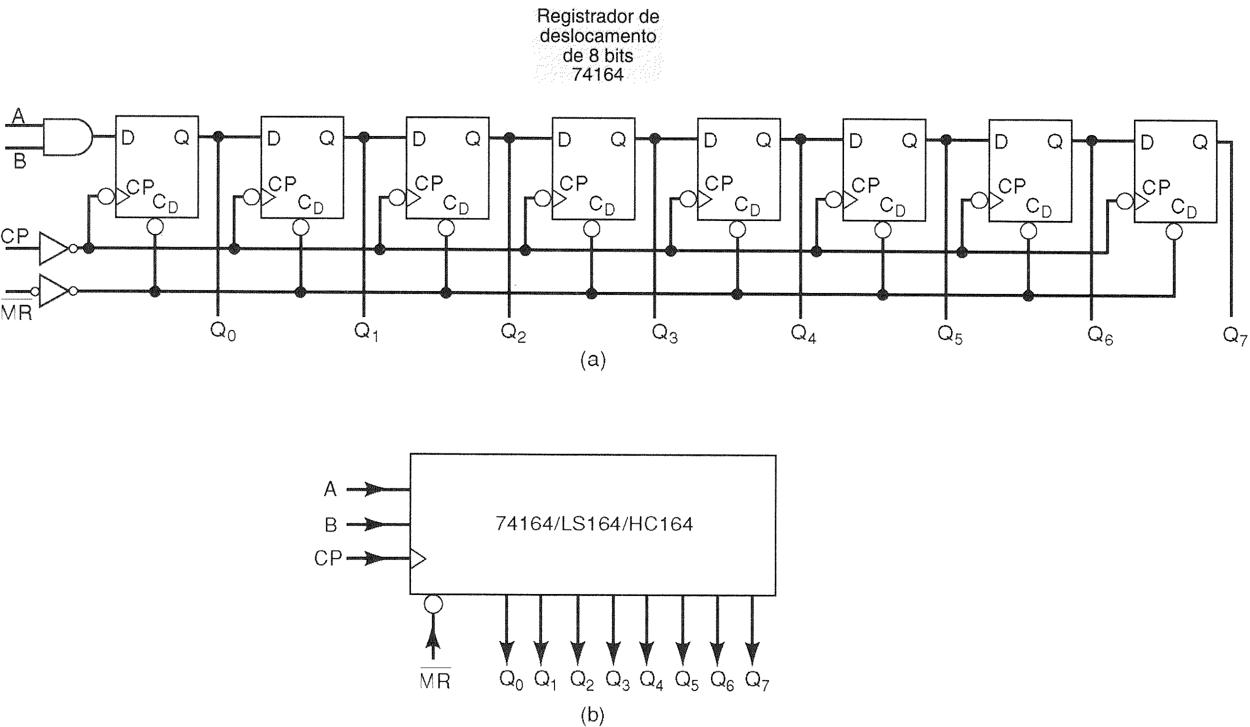
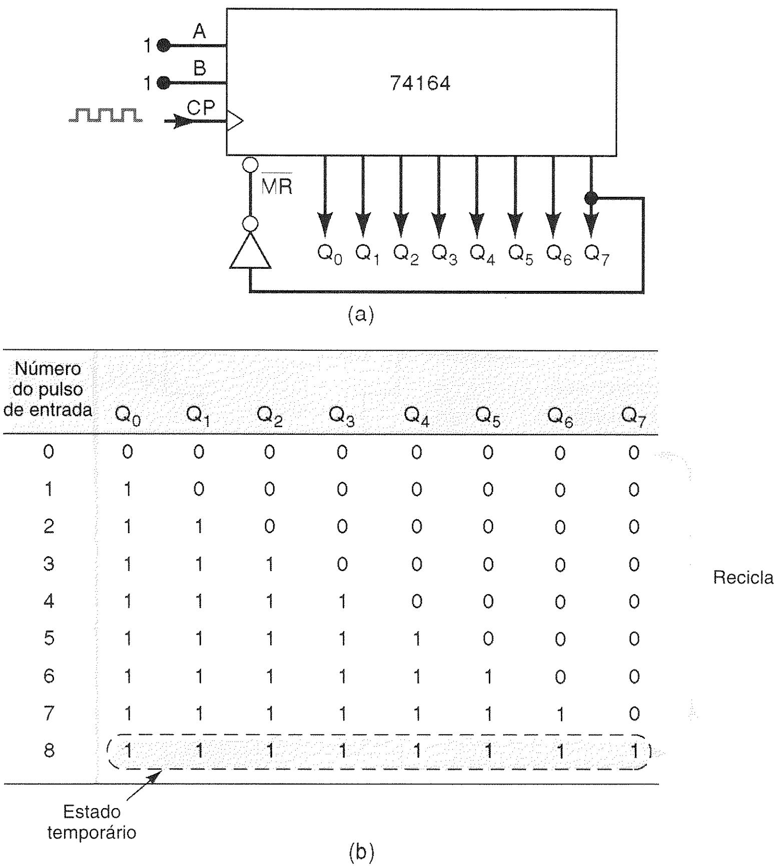


Fig. 7-56 - (a) Diagrama lógico para o 74164; (b) símbolo lógico. (Cortesia da Fairchild, uma companhia da Schlumberger.)



Número do pulso de entrada	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7
0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0
4	1	1	1	1	0	0	0	0
5	1	1	1	1	1	0	0	0
6	1	1	1	1	1	1	0	0
7	1	1	1	1	1	1	1	0
8	1	1	1	1	1	1	1	1

Fig. 7-57 - Exemplo 7-24.

Solução

A seqüência correta é dada na Fig. 7-57(b). Com $A = B = 1$, a entrada serial é 1, portanto 1s serão deslocados pelo registrador em cada subida de *CP*. Como Q_7 está inicialmente em 0, a entrada $\overline{MR}$ está inativa.

No oitavo pulso, o registrador tenta ir para o estado 11111111, quando o 1 de Q_6 se desloca para Q_7. Este estado ocorre apenas momentaneamente, pois $Q_7 = 1$ produz um nível BAIXO em $\overline{MR}$ que imediatamente resseta o registrador de volta para 00000000. Então a seqüência é repetida nos próximos oito pulsos de clock.

A seguir, está uma lista de alguns outros CIs registradores que são variações daqueles já apresentados:

- **74194/LS194/HC194.** Este é um CI *registrador de deslocamento bidirecional universal* de quatro bits que pode realizar as operações de deslocamento à esquerda, deslocamento à direita, entrada paralela e saída paralela. Estas operações são selecionadas por um código de seleção de modo de dois bits, aplicado como entrada para o dispositivo. O Problema 7-56 fornecerá a você uma oportunidade de descobrir mais sobre este chip versátil.
- **74373/LS373/HC373/HCT373.** Este é um registrador de oito bits (octal) de entrada paralela/saída paralela que contém oito latches do tipo *D* com saídas de *terceiro estado*. Uma saída de terceiro estado é um tipo especial de circuito lógico de saída que permite que as saídas do dispositivo sejam ligadas juntas de modo seguro. Estudaremos as características de dispositivos de terceiro estado, tal como o 74373, no próximo capítulo.
- **74374/LS374/HC374/HCT374.** Este é um registrador de oito bits (octal) de entrada paralela/saída paralela que contém oito flip-flops D, disparados pela borda, com saídas de terceiro estado.

Os CIs de registradores que foram apresentados aqui são representativos dos vários tipos disponíveis comercialmente. Embora existam muitas variações nestes registradores básicos, agora deve ser fácil de compreender a maioria deles a partir das folhas de características dos fabricantes.

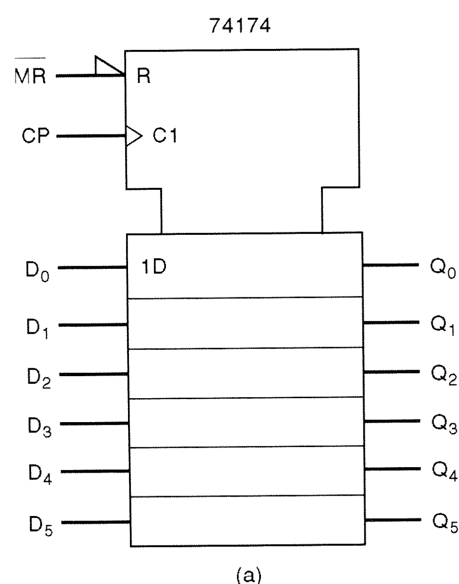
Apresentaremos diversas aplicações de registradores nos problemas do final do capítulo e no material abordado nos capítulos subseqüentes.

QUESTÕES DE REVISÃO

1. Que espécie de registrador pode ter um número binário completo carregado nele em uma operação, e então deslocá-lo um bit de cada vez?
2. *Verdadeiro ou falso:* Um registrador de entrada serial/saída paralela pode ter todos os seus bits mostrados de uma vez.
3. Que tipo de registrador pode ter entrada de dados de um bit de cada vez, mas tem todos os bits de dados disponíveis como saídas?
4. Em que tipo de registrador temos acesso apenas ao FF mais à esquerda ou mais à direita?
5. Como difere a entrada paralela de dados entre o 74165 e o 74178?
6. Como funciona a entrada *CP INH* do 74LS165?

7-23 SÍMBOLOS IEEE/ANSI PARA REGISTRADORES

Apresentaremos dois exemplos dos símbolos IEEE/ANSI para CIs de registradores. Primeiro, vamos considerar o CI 74174 de entrada paralela/saída paralela, cuja lógica interna e o símbolo lógico tradicional foram mostrados na Fig. 7-49. O símbolo IEEE/ANSI para o 74174 é dado na Fig. 7-58(a). Seu



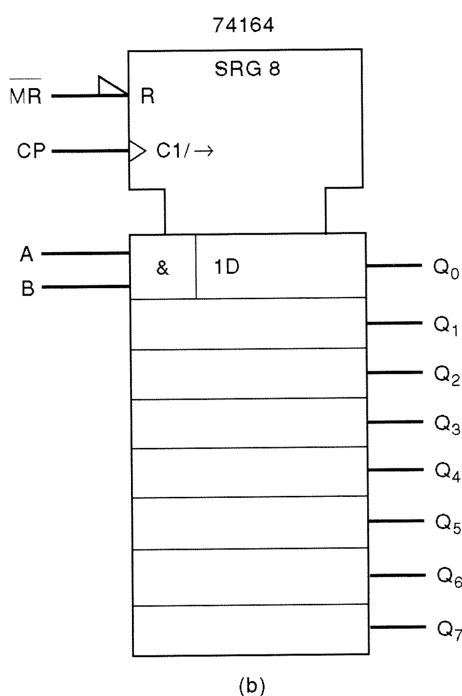


Fig. 7-58 - Símbolos IEEE/ANSI para (a) o 74174 registrador de entrada paralela/saída paralela e (b) o 74164 registrador de deslocamento de entrada serial/saída paralela.

contorno consiste no bloco de controle comum, com chanfro, e de seis retângulos estreitos representando os seis FFs.

O bloco de controle comum tem as entradas que são comuns a todos os elementos do CI; neste caso, as entradas $\overline{MR}$ e *CP* são comuns aos seis FFs, Q_0 até Q_5, que constituem o registrador. A indicação interna para a entrada $\overline{MR}$ é mostrada como um R para indicar que sua função é *ressetar* cada FF. A indicação interna para a entrada *CP* é C1, que nos informa que ela controla a entrada de dados em qualquer elemento de armazenamento que tiver um prefixo de 1 em seu nome de entrada. Cada entrada *D* dos FFs tem um indicador interno de 1D (mostrado apenas para Q_0, mas admite-se o mesmo para cada FF). O "1" em C1 e 1D estabelece a dependência das entradas *D* dos flip-flops com a entrada comum de clock *CP*.

O símbolo IEEE/ANSI para o registrador de deslocamento de entrada serial/saída paralela 74164 é apresentado na Fig. 7-58(b). Seu contorno consiste no bloco de controle comum e em oito FFs que formam o registrador. A notação SRG 8 identifica este CI como sendo um registrador de deslocamento de oito bits.

A entrada *CP* tem um indicador interno C1/→. A barra (/) é usada para separar as duas funções C1 e → realizadas por esta entrada. O C1 indica que *CP* controla a entrada de dados no flip-flop Q_0, já que Q_0 tem o indicador de entrada 1D. Note que o bit de dado que entra em Q_0 é a combinação AND das entradas *A* e *B*. Repare também que já que não existe entrada de dados externa para Q_1 até Q_7. *CP* não controla a entrada de dados destes FFs. A → denota que a transição ativa de *CP* produzirá a operação de deslocamento para a direita (de Q_0 em direção a Q_7).

QUESTÕES DE REVISÃO

1. O que significa a barra (/) quando ela aparece em um indicador de entrada?
2. Que notação deveria ser usada para descrever a função realizada por um dos registradores do CI 4731B da Fig. 7-53?

7-24 PESQUISA DE FALHAS

Flip-flops, contadores e registradores são os principais componentes em **sistemas de lógica seqüencial**. Um sistema de lógica seqüencial, devido aos seus dispositivos de memória, tem a característica de que suas saídas e as seqüências de operações dependem tanto das entradas presentes como das entradas anteriores. Embora os sistemas lógicos seqüenciais sejam geralmente mais complexos que os sistemas lógicos combinacionais, os procedimentos essenciais para pesquisar falhas se aplicam bem em ambos os tipos de sistemas. Sistemas seqüenciais sofrem dos mesmos tipos de falhas (circuitos abertos, em curto-circuito, falhas internas em CIs etc.) que os sistemas combinacionais.

Muitos dos passos utilizados para isolar as falhas em um sistema combinacional podem ser aplicados em sistemas seqüenciais. Uma das técnicas mais eficazes de depuração começa com o pesquisador observando a operação do sistema e, através de raciocínio analítico, determinando as causas possíveis do mau funcionamento do sistema. Então, ele utiliza os instrumentos de teste disponíveis para isolar o defeito. Os exemplos seguintes mostrarão o tipo de raciocínio analítico que seria o passo inicial na depuração de sistemas seqüenciais. Após estudar estes exemplos, você deveria estar pronto para "atacar" os problemas de depuração do fim do capítulo.

EXEMPLO 7-25

A Fig. 7-59(a) mostra um 74LS293 ligado como um contador de módulo 10. Um estudante testa a operação do contador aplicando um sinal de clock de 1 kHz e observa as saídas *Q* com um osciloscópio. As formas de onda são mostradas na Fig. 7-59(b). Determine as possíveis causas do comportamento incorreto do circuito.

Solução

As formas de onda mostram que Q_0 está comutando em resposta às descidas do clock, mas todos os outros FFs estão permanentemente no estado BAIXO. Diversas falhas possíveis poderiam produzir esta operação.

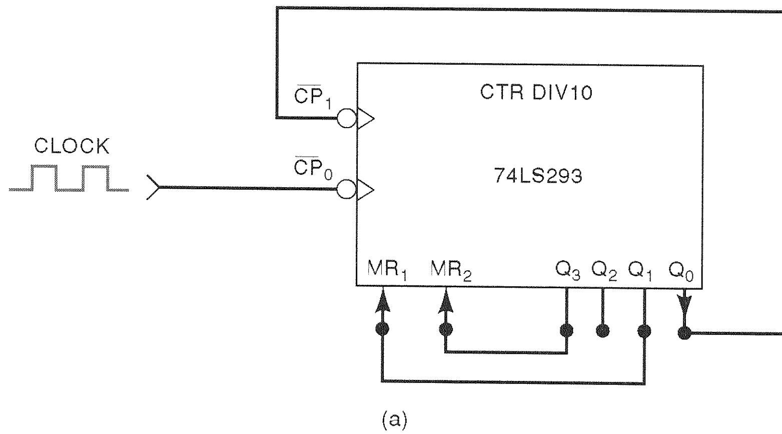
1. A saída Q_1 está interna ou externamente em curto com a terra. Consultando o diagrama do 74LS293 na Fig. 7-8(a), podemos ver que isto impediria Q_2 e Q_3 de comutar, tendo em vista que Q_1 é o sinal de clock para Q_2 e Q_2 é o sinal de clock para Q_3.
2. A conexão de Q_0 para $\overline{CP}_1$ está aberta, de modo que Q_1 não recebe sinal de clock.
3. Existe uma falha interna no CI que impede Q_1 de comutar.
4. A entrada MR_1 está internamente em curto-circuito com a terra, forçando $Q_1 = 0$.

EXEMPLO 7-26

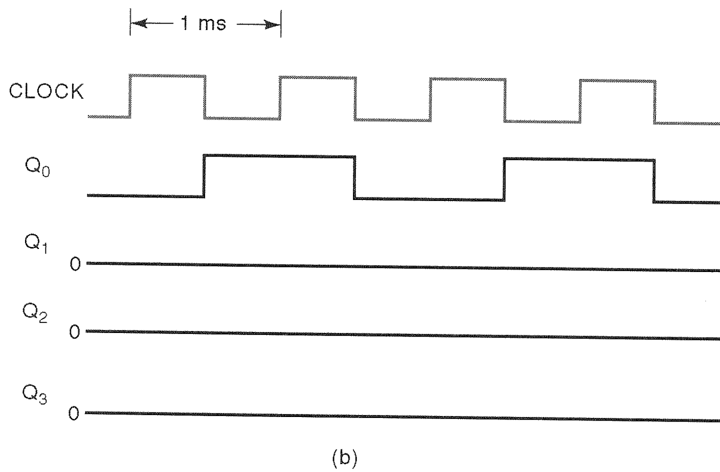
Após analisar a situação descrita no Exemplo 7-25, o estudante procedeu para isolar a falha. Ele realiza testes com o ohmímetro e verifica que Q_1 não está em curto com a terra e que Q_0 está conectado em $\overline{CP}_1$. Isto elimina as duas primeiras possibilidades de falha. Concluindo que o CI está ruim, ele o substitui. Para sua surpresa, a operação do circuito exibe o mesmo sintoma. Coçando a cabeça, ele decide observar melhor as formas de onda dos FFs utilizando uma escala de tempo de 10 ns/cm por divisão. Nesta escala, ele pode ver um glitch muito estreito ocorrendo no sinal Q_1 no instante da descida de Q_0 (vide Fig. 7-60). Qual é a falha provável?

Solução

Quando o contador está operando corretamente, deveria haver um glitch em Q_1 quando o contador vai para o estado 1010 (10), no qual os níveis ALTOs em Q_3 e Q_1 fazem as entradas *MR* limparem a contagem de volta para 0000. As formas de onda na Fig. 7-60, entretanto, não mostram Q_3 em ALTO quando o glitch em Q_1 ocorre. O defeito mais provável é uma conexão aberta em MR_2, já que isto seria interpre-



(a)



(b)

Fig. 7-59 - Exemplo 7-25.

tado como um nível ALTO constante pelo circuito integrado TTL. Assim, tão logo Q_1 vai para ALTO, as entradas *MR* ficam ambas em ALTO e o contador é ressetado para 0000.

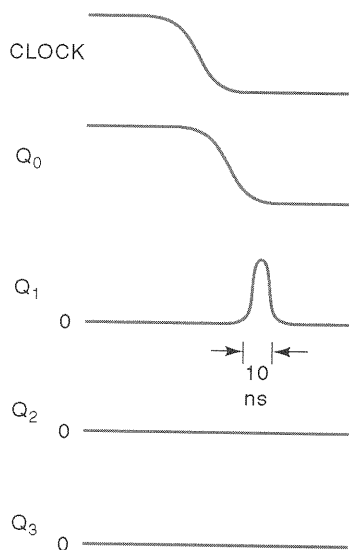


Fig. 7-60 - Exemplo 7-26.

EXEMPLO 7-27

Uma estudante testa o freqüencímetro da Fig. 7-46 para diversos intervalos de amostragem e para diferentes freqüências desconhecidas de entrada. Em todos os casos, ela encontra que a freqüência indicada é exatamente *duas vezes* o que deveria ser. Qual é a provável causa do mau funcionamento?

Solução

Consultando a Fig. 7-46, notamos que a freqüência desconhecida é habilitada, através da porta AND, no contador durante o intervalo de t_3 até t_4, enquanto SAMPLE e *X* estão ambos em nível ALTO. Se a entrada do meio da porta AND estiver aberta, funcionaria como se estivesse permanentemente em ALTO (admitindo-se dispositivos TTL). Isto permitiria que pulsos da freqüência desconhecida passassem pela porta, enquanto *X* fosse ALTO durante o intervalo de t_2 até t_4. Isto é duas vezes o intervalo normal, e, portanto, o contador contaria duas vezes o valor normal.

EXEMPLO 7-28

Um estudante monta o relógio digital das Figs. 7-47 e 7-48. Ele observa que a seção dos SEGUNDOS está contando

TABELA 7-7

Dezenas das HORAS	Unidades das HORAS	
0	1	←
0	2	
0	3	
0	4	
0	5	
0	6	recicla e repete
0	7	
1	8	
1	9	
1	0	
1	1	
1	2	

adequadamente. Para testar rapidamente a operação das seções dos MINUTOS e das HORAS, ele desconecta o contador de módulo 60 para que os contadores sejam acionados a uma taxa 60 vezes maior do que a normal. Ele observa que a seção dos MINUTOS está operando corretamente, mas a seção das HORAS conta e apresenta os números da maneira indicada na Tabela 7-7. Qual é a provável causa desta seqüência incorreta?

Solução

Tendo em vista que o problema se encontra na seção das HORAS, precisamos consultar a Fig. 7-48. A seqüência está correta, com a exceção de que o dígito das dezenas é incrementado de 0 a 1 quando o dígito das unidades vai de 7 para 8, em vez de quando vai de 9 para 0. Esta operação ocorreria se a entrada *CLK* do flip-flop *X* fosse erroneamente conectada em Q_2, no lugar de Q_3 do contador BCD. Se fosse este o caso, então, quando o contador BCD incrementasse de 7 para 8, seu flip-flop Q_2 geraria uma descida que comutaria o flip-flop *X* mais cedo do que o esperado.

RESUMO (PARTE II)

1. Um freqüencímetro é um circuito que utiliza contadores binários para medir e apresentar em um display a freqüência de um sinal de entrada.
2. Um circuito de relógio digital usa contadores binários para acompanhar e mostrar as horas do dia.
3. Numerosos CIs de registradores estão disponíveis e podem ser classificados de acordo com o tipo de entrada que possuem: paralelas (todos os bits apresentados simultaneamente), seriais (um bit de cada vez) ou ambas. Do mesmo modo, registradores podem ter saídas paralelas (todos os bits disponíveis simultaneamente) ou seriais (um bit de cada vez).
4. Um sistema de lógica seqüencial utiliza FFs, contadores e registradores, além de portas lógicas. Suas saídas e a seqüência de operações dependem das entradas passadas e atuais.
5. A pesquisa de falhas de um sistema de lógica seqüencial começa pela observação da operação do sistema, seguida de um raciocínio analítico para determinar as possíveis causas de algum mau funcionamento e, finalmente, de medidas de teste para isolar a falha real.

TERMOS IMPORTANTES (PARTE II)

freqüencímetro
intervalo de amostragem
entrada paralela/saída paralela
entrada serial/saída serial
entrada paralela/saída serial
entrada serial/saída paralela
sistema de lógica seqüencial

PROBLEMAS

PARTE I

SEÇÕES 7-1 E 7-2

7-1. Uma onda quadrada de 8 MHz aciona um contador assíncrono de 5 bits. Qual é a freqüência do último FF (MSB)? Qual é a taxa de ciclo da sua forma de onda de saída?

7-2. Repita o Problema 7-1 para uma entrada com taxa de ciclo de 20%.

7-3. Admita que o contador binário de 5 bits comece no estado 00000. Qual será a contagem depois de 144 pulsos de entrada?

D7-4. Utilize flip-flops J-K e qualquer outra lógica necessária para construir um contador assíncrono de módulo 24.

7-5. Desenhe as formas de onda para todos os FFs no contador decádico da Fig. 7-6(b) em resposta a uma freqüência de clock de 1 kHz. Mostre qualquer glitch que possa aparecer nas saídas dos FFs. Determine a freqüência da saída *D*.

7-6. Repita o Problema 7-5 para o contador da Fig. 7-6(a).

7-7. Altere as entradas para a porta NAND da Fig. 7-7 de modo que o contador divida a freqüência por 50. Repita para uma divisão de freqüência por 100.

7-8. Um contador, ou um grupo de contadores, é freqüentemente usado para dividir um sinal de clock de alta freqüência para uma freqüência de saída mais baixa. Quando estes contadores são contadores binários (isto é, eles contam na seqüência binária), a saída não será uma onda quadrada simétrica se a seqüência binária for encurtada para produzir o módulo desejado. Por exemplo, consulte a forma de onda *C* do contador de módulo 6 da Fig. 7-4.

Quando um contador está sendo usado apenas para divisão de freqüência, não é necessário que ele conte em uma seqüência binária, desde que ele tenha o módulo desejado. Uma onda quadrada simétrica de saída pode ser obtida para qualquer módulo *par* separando-se o módulo em um produto de dois módulos, um dos quais é uma potência de dois. Por exemplo, um contador de módulo 6 pode ser formado por um contador de módulo 3 e por um contador de módulo 2, como mostra a Fig. 7-61.

Nela, os flip-flops *A*, *B* e a porta NAND implementam o contador de módulo 3, cuja saída *B* tem um terço da freqüência dos pulsos de entrada. Esta saída *B* está conectada na entrada do flip-flop *C*, que está atuando como um contador de módulo 2, para dividir a freqüência para um sexto da freqüência de entrada.

(a) Admita que todos os FFs estão inicialmente em nível BAIXO e rascunhe as formas de onda para cada saída de FF por 12 ciclos da entrada.

(b) Construa o diagrama de transição de estados e mostre que esta não é uma seqüência binária normal.

SEÇÃO 7-3

D7-9. Mostre como um contador 74LS293 pode ser usado para produzir uma saída de 1,2 kpps a partir de uma entrada de 18 kpps.

Fig. 7-61 - Problema 7-8.

D7-10. Mostre como dois 74LS293 podem ser conectados para dividir uma freqüência de entrada por 60, produzindo uma onda quadrada de saída simétrica.

C7-11. Determine a freqüência na saída *X* da Fig. 7-62.

D7-12. (a) Acrescente a lógica necessária a um 74HC4024 de modo que ele opere como um contador de módulo 100.

Fig. 7-62 - Problemas 7-11 e 7-64.

(b) Utilize 74HC4024s e qualquer outra lógica necessária para converter um sinal de 10 kpps para 1 pps.

SEÇÃO 7-4

7-13. (a) Desenhe o diagrama para um contador *decrescente* de módulo 16.

(b) Construa o diagrama de transição de estados.

(c) Se o contador está inicialmente no estado 0110, em qual estado ele estará após 37 pulsos de clock?

7-14. Consulte o contador mostrado na Fig. 7-63. Como você pode saber que ele é um contador decrescente? Ele foi modificado, de modo que ele *não* conta a seqüência binária completa de 111 até 000. Determine a seqüência de contagem real que ele realiza.

SEÇÃO 7-5

7-15. Um contador por pulsação de quatro bits é acionado por um sinal de clock de 20 MHz. Desenhe as formas de onda das saídas de cada FF se cada FF tem t_{pd} = 20 ns. Determine que estados de contagem, caso existam, não ocorrerão por causa dos atrasos de propagação.

7-16. (a) Qual é a máxima freqüência de clock que pode ser usada com o contador do Problema 7-15?

(b) Qual seria f_{max} se o contador fosse expandido para seis bits?

SEÇÕES 7-6 E 7-7

7-17. (a) Desenhe o diagrama de circuito para um contador paralelo de módulo 64.

(b) Determine f_{max} para este contador se cada FF tem um t_{pd} = 20 ns e cada porta tem t_{pd} = 10 ns.

C7-18. A Fig. 7-64 mostra um contador paralelo de quatro bits que foi projetado de modo a não seguir os 16 estados binários. Analise sua operação determinando sua seqüência de contagem, e então desenhe as formas de onda para cada

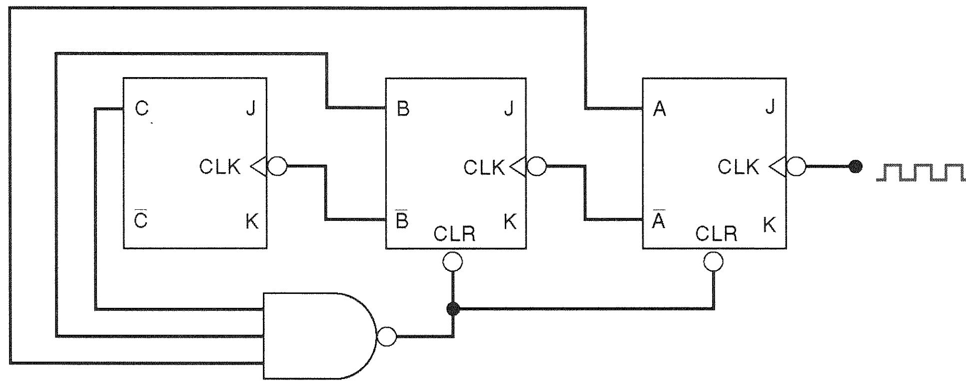


Fig. 7-63 - Problema 7-14.

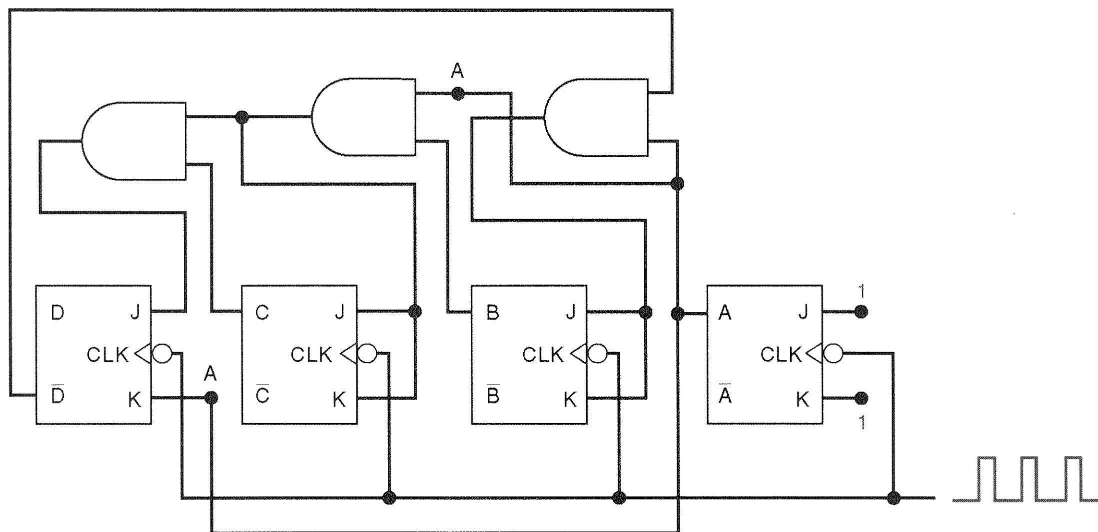


Fig. 7-64 - Problema 7-18.

saída. (Veja a Seção 5-23 para rever o procedimento de análise.) Admita que todos os FFs estão inicialmente no estado 0.

7-19. Desenhe o diagrama para um contador síncrono decrescente de módulo 8.

T7-20. Descreva como o contador crescente/decrescente da Fig. 7-18 operaria se a saída do INVERSOR ficasse sempre em ALTO.

SEÇÕES 7-8 E 7-9

C7-21. Mostre como conectar dois 74LS193s para formar um contador de 8 bits que divide a freqüência do clock por 100. Utilize a saída $\overline{TC}_D$ de um estágio como clock do segundo estágio. (*Sugestão:* O contador deve ser carregado quando ambos os estágios estiverem simultaneamente em suas contagens finais.)

N, C 7-22. A Fig. 7-65 mostra como um contador decrescente com carga pode ser usado em um circuito *temporizador programável.* A freqüência do clock de entrada é de 1 Hz, com precisão, derivada da freqüência da rede de 60 Hz, após a divisão por 60. As chaves S1 até S4 são usadas para ajustar o contador para uma contagem inicial quando um pulso é aplicado em $\overline{PL}$. A operação do temporizador é iniciada pressionando-se a chave de início. O flip-flop *Z* é usado para eliminar os efeitos da trepidação de contato da chave de INÍCIO. O MONO é usado para fornecer um pulso bem estreito para a entrada $\overline{PL}$. A saída do flip-flop *X* terá uma forma de onda que vai para ALTO por um número de segundos igual ao número presente nas chaves.

(a) Admita que todos os FFs e o contador estão no estado 0, e analise e explique a operação do circuito, mostrando formas de onda quando for necessário, para o caso em que SW1 e SW4 estão em BAIXO e SW2 e SW3 estão em ALTO. Não deixe de explicar a função do flip-flop *X*.

(b) Por que a saída do temporizador não pode ser tomada da saída $\overline{TC}_D$?

(c) Por que a chave de INÍCIO não pode ser usada para disparar o MONO diretamente?

(d) O que aconteceria se a chave de INÍCIO fosse mantida acionada por um longo período? Inclua a lógica necessária para garantir que a manutenção do acionamento da chave de INÍCIO não vai afetar a operação do temporizador.

7-23. Modifique o circuito da Fig. 7-24 de modo que ele funcione como um contador de módulo 10. A freqüência da saída Q_3 deveria ser um décimo da freqüência da entrada CP_D. Desenhe as formas de onda de Q_3, Q_2, Q_1, Q_0 e $\overline{TC}_D$.

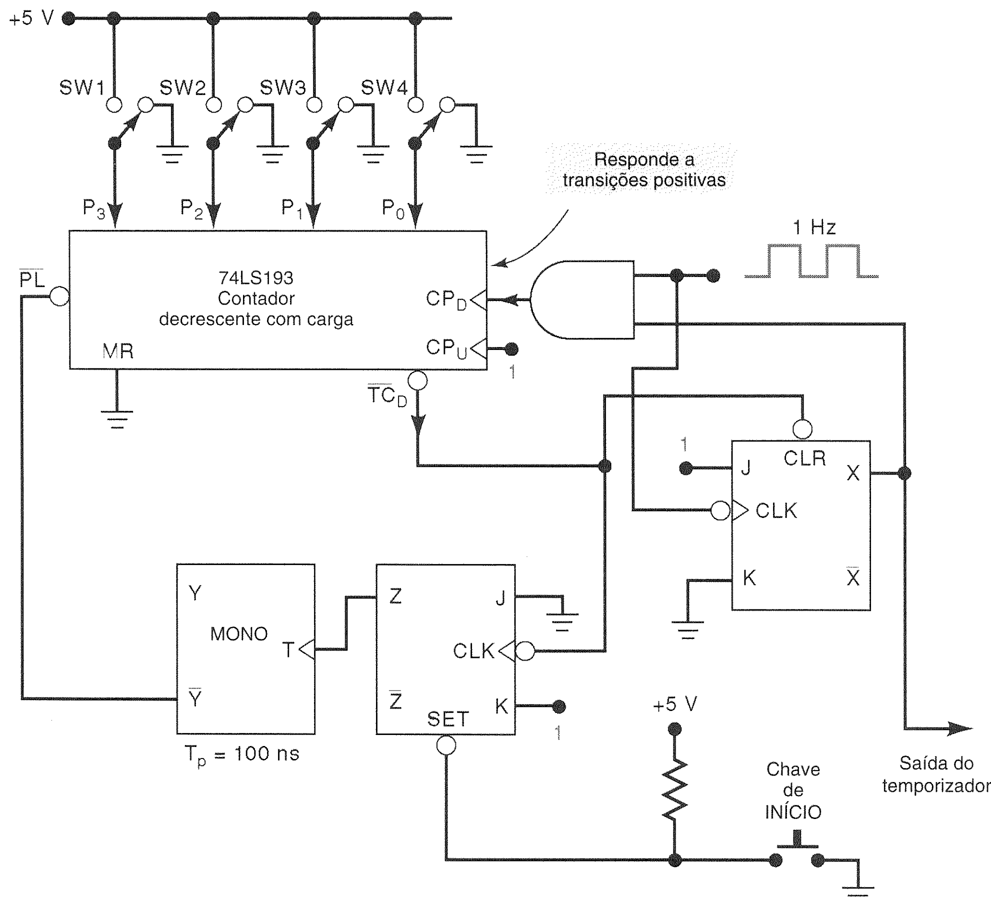


Fig. 7-65 - Problemas 7-22, 7-63 e 7-66.

7-24. Altere as entradas de dados paralelos na Fig. 7-24 para 1001. Desenhe as formas de onda em Q_3, Q_2, Q_1, Q_0 e $\overline{TC}_D$. Qual é o módulo?

7-25. Desenhe as formas de onda para os sinais de entrada necessários para realizar as seguintes seqüências de operações no circuito da Fig. 7-25: (1) limpar a contagem para 0; (2) contar crescente até 24_{10}; (3) carregar o contador com 76_{10}; (4) contar decrescente até 0.

SEÇÃO 7-10

7-26. A Fig. 7-66 mostra o símbolo IEEE/ANSI para um CI 7490 ou 74290. Examine o símbolo e determine o seguinte:

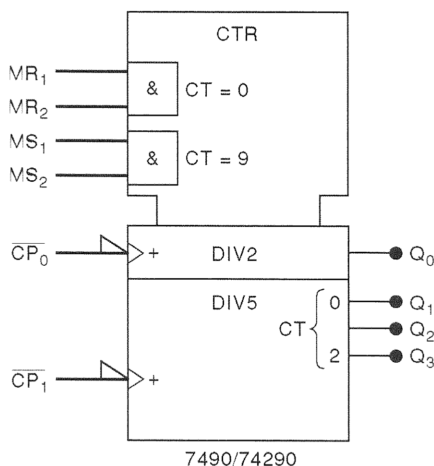


Fig. 7-66 - Problema 7-26.

(a) O módulo total
(b) A função realizada pelas entradas *MR*
(c) A função realizada pelas entradas *MS*
(d) Ele é um contador crescente ou decrescente?
(e) Como você o conectaria para que ele funcionasse como um contador BCD? (Consulte a folha de características no Apêndice.)
(f) Como você o conectaria para que ele dividisse a freqüência do clock por 10 e produzisse uma onda quadrada simétrica?

7-27. O CI contador 74192 opera exatamente como o 74193, exceto pelas seguintes diferenças:

- O 74192 é um contador BCD, que tanto conta de modo crescente, de 0 a 9, como de modo decrescente, de 9 a 0.
- A saída $\overline{TC}_U$ é ativada quando a contagem é 9 e a entrada CP_U está em BAIXO.

Modifique o símbolo IEEE/ANSI da Fig. 7-26 de modo que ela represente o 74192.

SEÇÕES 7-11 E 7-12

7-28. Desenhe as portas necessárias para decodificar todos os estados de um contador de módulo 16 usando saídas ativas em BAIXO.

7-29. Desenhe as portas AND necessárias para decodificar os 10 estados do contador BCD da Fig. 7-6(b).

7-30. A Fig. 7-67 mostra um contador assíncrono sendo usado para ajudar a gerar sinais de controle. Os sinais de controle 1 e 2 poderiam ser utilizados para muitos propósitos, incluindo o controle de motores, solenóides, válvulas e aquecedores. Determine as formas de onda de controle, admitindo que todos os FFs estejam inicialmente em BAIXO. Ignore os

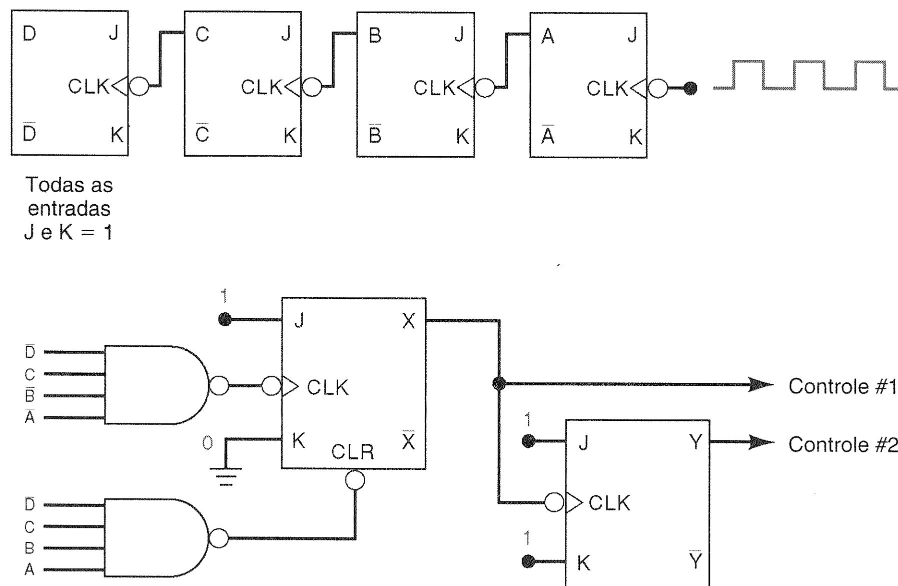


Fig. 7-67 - Problema 7-30.

glitches de decodificação. Admita que a freqüência de clock seja igual a 1 kpps.

7-31. Desenhe todas as formas de onda das saídas das portas de decodificação de um contador *por pulsação* de módulo 16, incluindo quaisquer glitches ou spikes que possam ocorrer devido aos atrasos dos FFs. Por que são as portas que estão decodificando números *pares* as únicas que têm glitches?

7-32. O circuito da Fig. 7-67 pode não funcionar bem por causa dos glitches nas saídas das portas NAND de decodificação.

(a) Determine em que ponto(s) os glitches podem causar problemas de operação.

(b) Quais são duas maneiras que podem ser usadas para eliminar a possibilidade de operação com erros?

SEÇÃO 7-13

7-33. Quantos FFs são usados na Fig. 7-32? Indique o estado de cada um destes FFs após 795 pulsos terem ocorrido.

7-34. Quantos contadores BCD em cascata são necessários para contar até 8000? Quantos FFs isto requer? Compare isto com o número de FFs necessários para um contador binário normal contar até 8000. Tendo em vista que ele usa mais FFs, por que o método de ligar contadores BCD em cascata é usado?

SEÇÃO 7-14

D7-35. (a) Projete um contador síncrono que realiza a seguinte seqüência: 000, 010, 101, 110, e repete. Os estados indesejáveis (não-usados) 001, 011, 100 e 111 devem sempre levar para 000 no PRÓXIMO pulso de clock.

(b) Projete novamente o contador da parte (a) sem nenhuma restrição sobre os estados não-usados; isto é, seus PRÓXIMOS estados podem ser *don't cares*. Compare com o projeto feito em (a).

C, D 7-36. Utilize o procedimento de projeto de contadores síncronos para projetar um contador síncrono decrescente de quatro bits que conta por todos os estados desde 1111 até 0000. Compare o resultado com o contador síncrono decrescente descrito na Seção 7-7.

C, D 7-37. Utilizando um procedimento similar ao que foi seguido no projeto do contador para acionar o motor de passo (Fig. 7-39), projete um contador síncrono de três bits que conta crescente ou decrescente sob o controle de uma entrada de Direção, *D*. Ele deve contar de modo crescente quando $D = 1$, e contar decrescente quando $D = 0$. (*Sugestão:* Este é um problema de *quatro* variáveis.)

Compare seu circuito final com o contador síncrono crescente/decrescente da Fig. 7-18.

SEÇÃO 7-15

7-38. Desenhe o diagrama para o contador em anel de cinco bits utilizando flip-flops J-K.

7-39. Combine o contador em anel do Problema 7-38 com um *único* flip-flop J-K para produzir um contador de módulo 10. Determine a seqüência de estados para este contador. Este é um exemplo de um contador decádico que não é um contador BCD.

7-40. Desenhe o diagrama para um contador Johnson de módulo 10 utilizando flip-flops J-K, e determine sua seqüência de contagem. Desenhe o circuito decodificador necessário para decodificar cada um dos 10 estados. Este é um outro exemplo de contador decádico que não é um contador BCD.

7-41. Determine a freqüência dos pulsos nos pontos *w*, *x*, *y* e *z* no circuito da Fig. 7-68.

7-42. (a) Um grupo de oito luzes de sinalização em uma máquina de jogos é controlado pelos FFs de um contador em anel de 8 bits que está sendo acionado por um sinal de clock de 2 pps. Descreva o efeito visual que é produzido.

(b) Repita para um contador Johnson de oito bits.

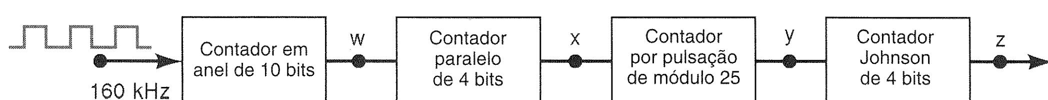


Fig. 7-68 - Problema 7-41.

PARTE II

SEÇÃO 7-16

7-43. Conforme foi salientado no texto, o freqüencímetro da Fig. 7-46 tem a desvantagem de o display mostrar todas as operações do contador (reset, contagem, manutenção), e portanto poderia ficar confuso, ou até mesmo ilegível. Isto pode ser resolvido pela inclusão de registradores buffers para armazenar a contagem no final de cada intervalo (t_3 a t_4 na Fig. 7-46) e mantê-lo para o display até o final do próximo intervalo de contagem (t_7 a t_8). A Fig. 7-69 mostra esta modificação. Um registrador formado por quatro flip-flops D foi inserido entre cada contador BCD e sua unidade decodificador/display.

(a) Analise este circuito e descreva sua operação, particularmente a transferência de dados dos contadores para o display.

(b) O que você veria em um display de três dígitos se a freqüência desconhecida estivesse constante em 2.570 pps e o intervalo de amostragem fosse 0,1 s?

(c) O que você veria neste display se a freqüência desconhecida fosse subitamente alterada para 3.230 pps?

7-44. O freqüencímetro da Fig. 7-69 utiliza três contadores BCD e um intervalo de amostragem de 100 microssegundos. Determine as leituras nos três displays do freqüencímetro para cada uma das seguintes freqüências de entrada.

(a) 220 kpps

(b) 4,5 Mpps

(c) 750 pps

SEÇÃO 7-17

7-45. Projete o circuito completo para a seção de SEGUNDOS do circuito do relógio digital da Fig. 7-47. Utilize um 74LS293 para o contador de módulo 6 e um 74LS290 para o contador BCD (veja a folha de características no Apêndice).

D7-46. O relógio digital da Fig. 7-47 deve ter algum meio de ajustar as seções de HORAS e MINUTOS para o tempo inicial correto. Por exemplo, isto pode ser feito levando o sinal de 1 pps para a entrada da seção de MINUTOS quando um botão de AJUSTE MINUTOS é ativado. Uma operação similar pode ser feita com um botão de AJUSTE HORAS. Projete a lógica necessária para implementar esta característica utilizando duas chaves do tipo botão.

D7-47. Modifique a seção das HORAS do relógio digital (Fig. 7-48) de modo que ele conte e mostre o tempo do modo militar (isto é, de 00 a 23 horas).

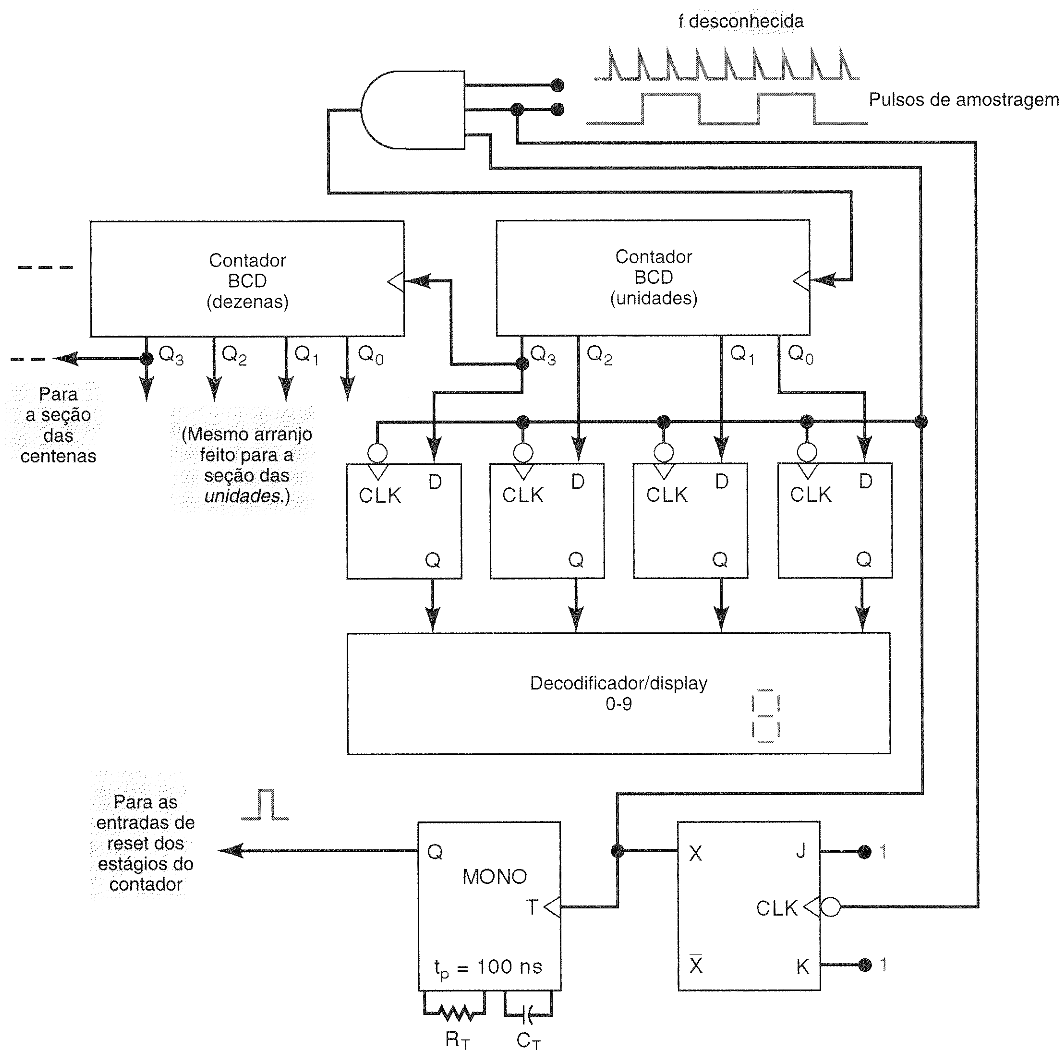


Fig. 7-69 - Problemas 7-43 e 7-48.

SEÇÕES 7-19 E 7-20

D7-48. Modifique o freqüencímetro da Fig. 7-69 de modo que ele utilize CIs 74174. Considere que o freqüencímetro contem três contadores BCD e um display de três dígitos.

D7-49. Mostre como conectar dois 74178s como um contador em anel de oito bits.

N, C7-50. A Fig. 7-70 mostra como um 74178 pode ser usado como um *conversor paralelo-série*. Os dados paralelos, que são colocados de P_0 até P_3, são deslocados serialmente para fora, de modo que uma forma de onda serial aparece em Q_3. O deslocamento para a saída dos dados paralelos é controlado pela ocorrência do pulso de INÍCIO. Este pulso de INÍCIO ocorre assincronamente em relação aos pulsos de clock, e portanto os flip-flops D são usados para sincronizar o carregamento paralelo e o deslocamento no 74178.

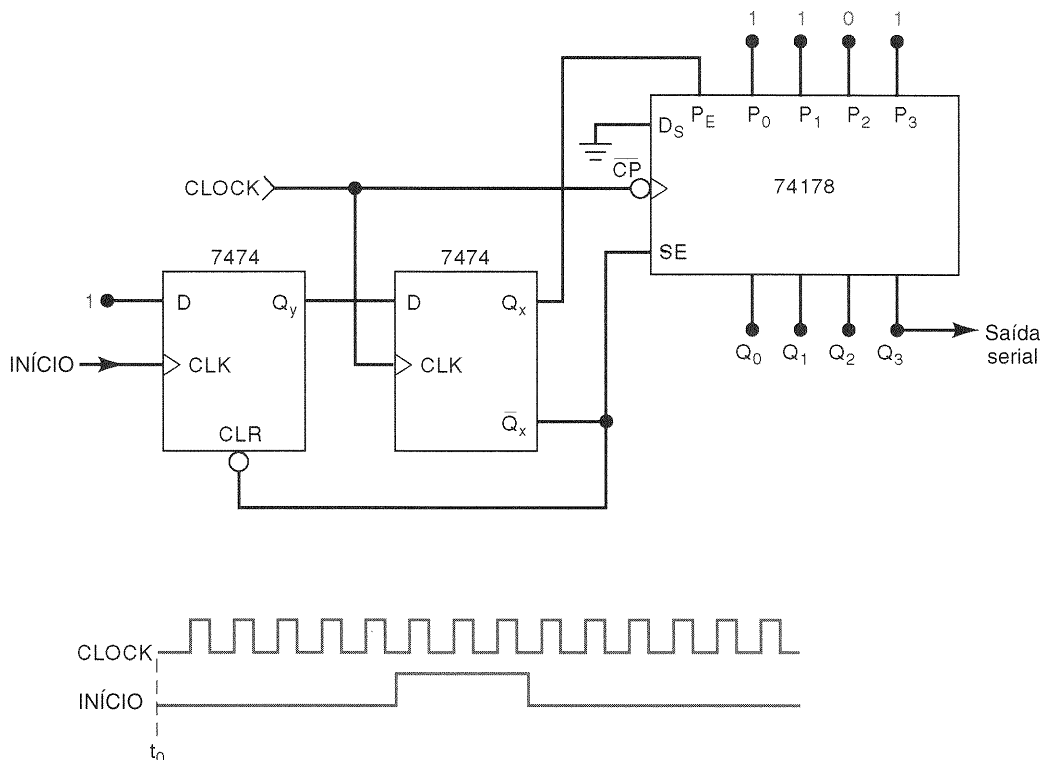


Fig. 7-70 - Problema 7-50.

Admita que o pulso de INÍCIO estava em BAIXO e que pulsos de clock foram continuamente aplicados ao circuito por um longo período de tempo antes do tempo t_0 (veja as formas de onda). Desenhe as formas de onda que aparecem em Q_X, $\overline{Q}_X$, Q_Y e Q_3 em resposta ao pulso de INÍCIO mostrado na figura.

D 7-51. O 74LS174 é um registrador de entrada paralela/saída paralela. No entanto, ele pode ser conectado de modo a funcionar como um registrador de deslocamento (entrada serial/saída serial). Mostre como isto pode ser feito. Então, mostre como, com a inclusão da lógica necessária, ele pode funcionar como um contador Johnson.

7-52. Considere a situação apresentada pelas formas de onda na Fig. 7-54. Se D_S for para BAIXO imediatamente antes de t_{247}, quando Q_{63} irá para BAIXO?

7-53. Mostre como o chip 4731B pode ser conectado como um registrador de deslocamento de 256 bits.

SEÇÕES 7-21 E 7-22

C, D 7-54. Modifique o circuito da Fig. 7-57 de modo que a saída do INVERSOR fique conectada na entrada *A*, em vez de conectada em $\overline{MR}$.

(a) Desenhe as formas de onda de saída de cada FF em resposta às formas de onda de entrada mostradas na Fig. 7-71.

(b) Adicione a lógica necessária para produzir um sinal de saída que vai para ALTO somente durante os intervalos de tempo de t_1 até t_2 e de t_8 até t_9.

(c) Adicione a lógica necessária para produzir um sinal que vai para BAIXO somente durante o intervalo de tempo de t_4 até t_7.

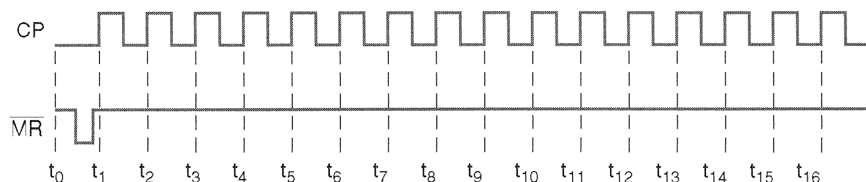


Fig. 7-71 - Problema 7-54.

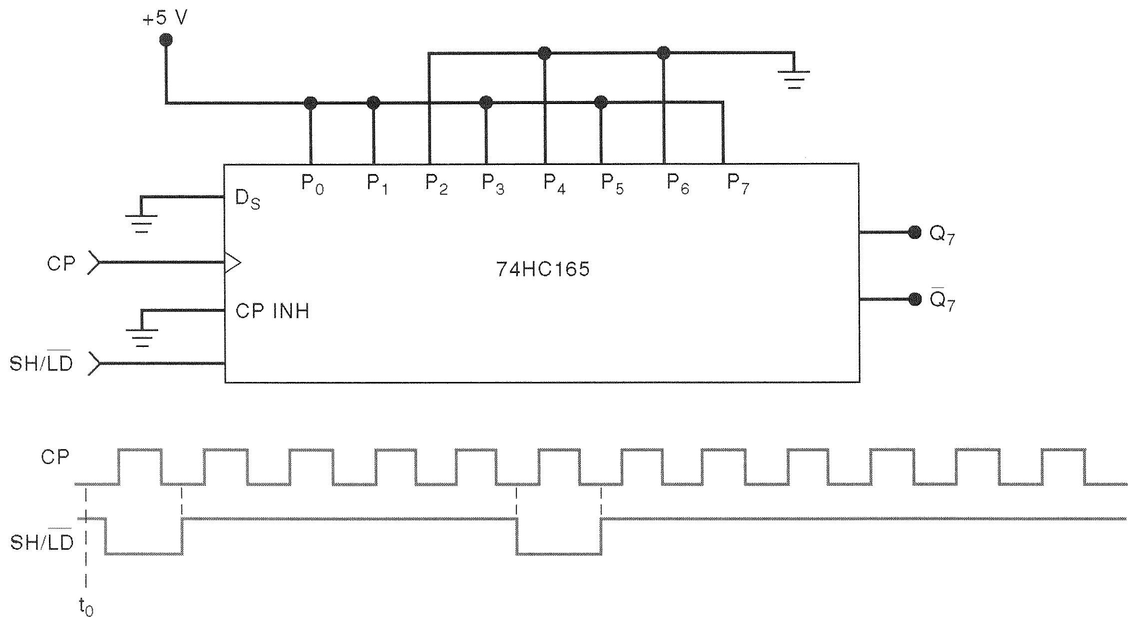


Fig. 7-72 - Problema 7-55.

7-55. Um 74HC165 está conectado conforme mostrado na Fig. 7-72. Admita que antes de t_0 pulsos foram aplicados em *CP*, e que a entrada $SH/\overline{LD}$ foi mantida em nível ALTO por um longo tempo. Desenhe a forma de onda de Q_7 em resposta às formas de onda de *CP* e $SH/\overline{LD}$ começando em t_0.

N 7-56. Enquanto examina o diagrama esquemático de uma parte de um equipamento, um estudante ou um engenheiro quase sempre se deparam com um CI que não é familiar. Em tais casos, freqüentemente é necessário consultar o manual do fabricante do CI para buscar especificações sobre o dispositivo. As informações na folha de características do CI são sempre completas, mas por vezes é difícil compreendê-las, especialmente por alguém com muito pouca experiência. Este problema vai dar a você prática em obter informações sobre um CI razoavelmente complexo — o 74194 registrador de deslocamento universal bidirecional. Consulte um manual TTL ou CMOS para responder as questões a seguir. Justifique suas respostas.

(a) A entrada *CLR* é síncrona ou assíncrona?

(b) *Verdadeiro ou falso:* Quando *CLK* está em BAIXO, os níveis de S_0 e S_1 não têm efeito algum sobre o registrador.

(c) Admita as seguintes condições:

$$Q_AQ_BQ_CQ_D = 1\ 0\ 1\ 1$$
$$ABCD = 0\ 1\ 1\ 0$$
$$\overline{CLR} = 1$$
$$\text{SR SER} = 0$$
$$\text{SL SER} = 1$$

Se $S_0 = 0$ e $S_1 = 1$, quais serão as saídas do registrador após um pulso *CLK*? E depois de dois pulsos *CLK*? E depois de três? Depois de quatro?

(d) Utilize as mesmas condições, exceto que $S_0 = 1$ e $S_1 = 0$, e repita a parte (c).

(e) Repita a parte (c) com $S_0 = S_1 = 1$.

(f) Repita a parte (c) com $S_0 = S_1 = 0$.

(g) Utilize as mesmas condições como na parte (c), mas considere que a saída Q_A está conectada em SL SER. Quais serão as saídas do registrador após quatro pulsos *CLK*?

(h) Mostre como conectar este CI para funcionar como um contador em anel, que conta através da seqüência Q_A Q_B Q_C Q_D: 0001, 0010, 0100, 1000, e repete.

SEÇÃO 7-24

T7-57. Um estudante testa o contador da Fig. 7-59(a) aplicando um sinal de clock de baixa freqüência e monitorando as saídas dos FFs em LEDs indicadores. Ele observa uma seqüência repetitiva indicada pelos LEDs (Tabela 7-8). Quais são as razões possíveis para o contador não estar contando adequadamente?

T7-58. Consulte o circuito do relógio digital das Figs. 7-47 e 7-48. Um estudante testando o circuito observou que as seções dos SEGUNDOS e dos MINUTOS contavam adequadamente, mas a seção das HORAS contava o seguinte: 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 11, 12, 11, 12,... Qual é a causa provável para este mau funcionamento?

T7-59. Um estudante testa o circuito de relógio digital (Figs. 7-47 e 7-48) e observa que a seção das HORAS não conta e que a seção dos MINUTOS conta de 00 até 39, e então recicla para 00 e repete. Quais são as possíveis causas deste comportamento incorreto?

T7-60. Consulte o freqüencímetro modificado da Fig. 7-69. Admita que existam três estágios de contadores BCD com seus respectivos registradores buffer. O intervalo de amostragem é ajustado para 1 segundo, e a freqüência desconhecida é 125 pps. Descreva o que aparecerá no display para cada uma das seguintes falhas do circuito:

TABELA 7-8

Q_3	Q_2	Q_1	Q_0	
0	0	0	0	←
0	0	0	1	
0	0	1	0	
0	0	1	1	recicla e
0	1	0	0	repete
0	1	0	1	
0	1	1	0	
0	1	1	1	

(a) Uma conexão aberta na entrada superior da porta AND
(b) Um resistor R_T queimado

T7-61. Uma estudante testa o freqüencímetro da Fig. 7-69 utilizando um intervalo de amostragem de 1 s e uma freqüência de 125 pps. Ela espera ver um display com 125, mas em vez disso vê um display que muda de poucos em poucos segundos como se segue: 125, 250, 375, 500, 625, 750, 875, 000, 125, 250, ... Qual pode ser a causa deste mau funcionamento?

T7-62. Consulte o contador crescente/decrescente da Fig. 7-18. Descreva como cada uma das falhas no circuito relacionadas a seguir afetará as operações de contagem crescente e decrescente.
(a) A saída da porta AND 4 está internamente em curto-circuito com V_{CC}.
(b) Uma ponte de solda está conectando a saída da porta AND 1 com a saída da porta AND 3.

T7-63. Um estudante realiza um teste no circuito temporizador da Fig. 7-65 e registra os resultados mostrados na Tabela 7-9. Examine os dados registrados e determine as causas possíveis para a operação com falhas.

T7-64. Um estudante montou o circuito contador da Fig. 7-62. Ele aplica um sinal preciso de 8,64 kpps na entrada e mede a freqüência de 54 pps em *X*, em vez dos esperados 60 pps. Qual é o erro de ligação mais provável que ele cometeu?

QUESTÕES DE FIXAÇÃO

7-65. Para cada uma das sentenças a seguir, indique o(s) tipo(s) de contador(es) que está(ão) sendo descrito(s).
(a) Cada FF é disparado pelo clock ao mesmo tempo.
(b) Cada FF divide a freqüência na sua entrada *CLK* por 2.
(c) A seqüência de contagem é 111, 110, 101, 100, 011, 010, 001, 000.
(d) O contador tem 10 estados distintos.
(e) O atraso total de comutação é a soma dos atrasos individuais dos FFs.
(f) Este contador não requer lógica de decodificação.
(g) O módulo do contador é sempre duas vezes o número de FFs.
(h) Este contador divide a freqüência de entrada pelo seu módulo.
(i) Este contador pode começar sua seqüência de contagem em qualquer estado inicial desejado.
(j) Este contador pode contar em ambas as direções.
(k) Pode sofrer de glitches de decodificação.
(l) Conta de 0 até 99.
(m) Pode ser projetado para contar por seqüências arbitrárias, determinando-se a lógica necessária para cada entrada *J* e *K* dos flip-flops.

APLICAÇÃO EM MICROCOMPUTADOR

C, D 7-66. Um microprocessador que é usado em uma aplicação de controle deve freqüentemente controlar a temporização de eventos externos, tais como o ligar e desligar de dispositivos como solenóides, motores e relés. Tais ações podem ser temporizadas utilizando software e executando-se repetidamente um trecho de programa um determinado número de vezes. Isto, no entanto, impõe uma sobrecarga de trabalho na MPU, já que ela não pode fazer outra coisa enquanto está executando este procedimento. Por essa razão, a maioria dos intervalos temporizados é usualmente gerada por hardware, que está *sob controle da MPU*. Em outras palavras, a MPU enviará dados para o hardware para informar que intervalo de tempo deve ser gerado.

No Problema 7-22, vimos como o CI 74LS193 poderia ser utilizado em um circuito temporizador (Fig. 7-65) para gerar intervalos de tempo precisos correspondentes ao dado binário de quatro chaves. Este circuito pode ser modificado de maneira a que os dados binários venham da MPU, em vez de virem das chaves. Na Seção 5-20, vimos como uma MPU poderia transferir dados para um dispositivo externo utilizando suas saídas de endereços, dados e clock (Fig. 5-50).

Mostre como combinar estes dois circuitos para que a saída do temporizador *X* gere um nível ALTO por um intervalo de tempo (em segundos) igual ao número binário que a MPU coloca no contador 74LS193. Você pode eliminar qualquer circuito que não seja necessário. Considere que o sinal *CP* da MPU é uma onda quadrada de 1 MHz. Lembre-se de que $\overline{PL}$ é uma entrada assíncrona.

TABELA 7-9

S1	S2	S3	S4	Saída(s) do Temporizador
5 V	0 V	0 V	0 V	10
0 V	0 V	5 V	5 V	3
5 V	0 V	5 V	5 V	11
5 V	5 V	5 V	0 V	14
0 V	5 V	0 V	5 V	7
5 V	5 V	0 V	0 V	14

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

PARTE I

SEÇÃO 7-1
1. Falso **2.** 0000 **3.** 128

SEÇÃO 7-2
1. *D*, *C* e *A* **2.** Verdadeiro, tendo em vista que um contador BCD tem 10 estados distintos. **3.** 5 kHz

SEÇÃO 7-3
1. 250 Hz **2.** $f_{in}/60$ **3.** 4.096
4. O contador tem módulo 64 e divide a freqüência por 64.
5. 96

SEÇÃO 7-4
1. Em um contador crescente, a contagem é incrementada de 1 a cada pulso de clock; em um contador decrescente, a contagem é decrementada de 1 com cada pulso.
2. A saída invertida de cada FF é conectada na entrada *CLK* do FF seguinte.

SEÇÃO 7-5
1. Cada FF adiciona seu atraso de propagação ao atraso total do contador em resposta ao pulso de clock.
2. Módulo 256

SEÇÃO 7-6
1. Pode operar em freqüências mais altas e tem circuitos mais complexos
2. Seis FFs e quatro portas AND
3. *ABCDE*

SEÇÃO 7-8
1. Ele pode ser ajustado para qualquer contagem inicial desejada.
2. A carga assíncrona é independente da entrada de clock, enquanto a carga síncrona ocorre na borda ativa do sinal de clock.

SEÇÃO 7-9

1. Quando $\overline{PL}$ é pulsado em BAIXO, o contador é carregado com o número binário presente nas suas entradas P_0 até P_3.
2. Um nível ALTO em *MR* sobrepõe-se a todas as outras entradas para ressetar o contador para 0000.
3. Verdadeiro **4.** 1, 1, 0, respectivamente **5.** 0 a 65.535

SEÇÃO 7-10

1. Veja o texto apropriado.
2. (a) Operação de contador crescente.
(b) Esta entrada é combinada por uma porta AND com qualquer outra entrada ou saída que tenha um "4" no seu identificador.
(c) Esta entrada controla o efeito de qualquer entrada que tenha "5" no seu identificador.
(d) Entrada de dados que é controlada pela entrada identificada por C5.

SEÇÃO 7-11

1. Sessenta e quatro
2. Uma porta NAND de seis entradas com entradas A, B, C, $\overline{D}$, E e $\overline{F}$.

SEÇÃO 7-12

1. Os glitches poderiam ser causados pelas mudanças de estado dos FFs, um de cada vez, durante as transições de estado do contador.
2. O sinal de strobe inibe as portas de decodificação até que todos os FFs tenham completado suas transições.

SEÇÃO 7-14

1. Veja o texto.
2. Ela mostra os níveis necessários em J e K para produzir todas as transições de estado possíveis dos FFs.
3. Ela mostra os níveis necessários em cada entrada J e K dos flip-flops para produzir as transições de estado do contador.
4. Verdadeiro

SEÇÃO 7-15

1. Contador em anel **2.** Contador Johnson
3. A saída invertida do último FF é conectada na entrada do primeiro FF.
4. (a) Falso **(b)** Verdadeiro **(c)** Verdadeiro

PARTE II

SEÇÃO 7-16

1. 1 ms **2.** Contador limpo; contador conta pulsos durante o intervalo de amostragem; contador pára e mantém a contagem para o display.
3. Um contador em anel utiliza mais FFs do que um contador Johnson.

SEÇÃO 7-17

1. Conformador de pulso, divisor de freqüência, contador de segundos e display, contador de minutos e display, contador de horas e display
2. Para mudar a transição de descida fornecida pela seção dos MINUTOS para uma subida necessária para o 74192

SEÇÃO 7-18 A SEÇÃO 7-22

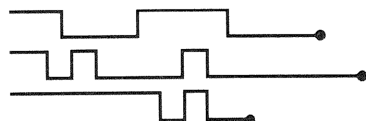
1. Entrada paralela/saída serial
2. Verdadeiro
3. Entrada serial/saída paralela
4. Entrada serial/saída serial
5. O 74165 utiliza transferência paralela assíncrona; o 74178 utiliza transferência síncrona.
6. Um nível ALTO inibe o deslocamento a cada *CP*.

SEÇÃO 7-23

1. Ela separa as duas funções indicadas e realizadas por aquela entrada.
2. SRG 64

CAPÍTULO 8

Famílias Lógicas de Circuitos Integrados



■ SUMÁRIO

OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Ler e compreender a terminologia de CIs digitais conforme especificação das folhas de características dos fabricantes.
- Comparar as características da série TTL padrão com as outras séries TTL.
- Determinar o fan-out para um determinado dispositivo lógico.
- Utilizar dispositivos lógicos com saídas do tipo coletor aberto.
- Analisar circuitos que contêm dispositivos tristate.
- Comparar as características das diversas séries CMOS.
- Analisar circuitos que usam chaves bilaterais CMOS para possibilitar que um sistema digital controle sinais analógicos.
- Descrever as principais diferenças e características das famílias lógicas TTL, ECL, MOS e CMOS.
- Relacionar e implementar as diversas considerações que são necessárias quando interligamos circuitos digitais de famílias lógicas diferentes.
- Usar comparadores de tensão para permitir que um sistema digital seja controlado por sinais analógicos.
- Utilizar um pulsador lógico e um rastreador de corrente como ferramentas para a pesquisa de falhas em circuitos digitais.

INTRODUÇÃO

Conforme descrito no Cap. 4, a tecnologia dos CIs digitais avançou rapidamente da integração em pequena escala (SSI), com menos de 12 portas por chip; para a integração em média escala (MSI), com 12 a 99 portas equivalentes por chip; até a integração em larga escala e a integração em escala muito ampla (LSI e VLSI), que pode ter dezenas de milhares de portas por chip; e, mais recentemente, para ULSI com mais de 100.000 portas por chip, e GSI com um milhão de portas ou mais.

A maioria das razões para os sistemas digitais modernos utilizarem circuitos integrados é óbvia. CIs contêm muito mais circuitos em um pequeno encapsulamento, de modo que o tamanho total de quase todos os sistemas digitais é reduzido. O custo diminui drasticamente devido às economias da produção em massa de grandes quantidades de dispositivos similares. Algumas das outras vantagens não são tão aparentes.

Os CIs tornam os sistemas digitais mais confiáveis reduzindo o número de conexões externas de um dispositivo para outro. Antes de termos os CIs, cada conexão do circuito era de um componente discreto (transistor, diodo, resistor etc.) para outro. Agora a maioria das conexões é interna aos CIs, onde estão protegidas de soldas ruins, interrupções ou curtos nas trilhas da placa e outros problemas físicos. Os CIs também reduziram drasticamente a potência elétrica necessária para realizar uma determinada função, já que seus circuitos miniaturizados tipicamente requerem menos potência que os equivalentes discretos. Além dos ganhos no custo da fonte de alimentação, esta redução na potência também significa que o sistema não necessita de muita ventilação.

Existem algumas coisas que os CIs não podem fazer. Eles não suportam correntes ou tensões muito grandes, pois o calor gerado em espaços tão pequenos causaria um aumento de temperatura acima dos limites aceitáveis. Além disso, não se pode implementar facilmente em CIs certos dispositivos elétricos, tais como indutores, transformadores e grandes capacitores. Por estas razões, os CIs são usados principalmente para realizar operações em circuitos de baixa potência, que são comumente denominadas *processamento de informação.* As operações que necessitam de maiores níveis de potência ou de dispositivos que não podem ser integrados ainda são realizadas com componentes discretos.

Com a vasta utilização dos CIs, veio a necessidade de saber e compreender as características elétricas das famílias lógicas dos CIs mais comuns. Lembre-se de que as diversas famílias lógicas diferem na maioria dos componentes que usam em seus circuitos. TTL e ECL utilizam transistores *bipolares* como principal elemento de circuito; PMOS, NMOS e CMOS usam transistores *MOS*FET como componente principal. Neste capítulo, apresentaremos as características importantes de cada uma destas famílias de CIs e de suas subfamílias. Uma vez que isto seja entendido, você estará bem mais preparado para fazer análise, pesquisa de falhas e projetar alguns circuitos digitais que contenham qualquer combinação destas famílias de CIs.

8-1 TERMINOLOGIA DE CIs DIGITAIS

Embora existam muitos fabricantes de CIs, a maior parte da nomenclatura e da terminologia é razoavelmente padronizada. Os termos mais úteis são definidos e apresentados a seguir.

Parâmetros de Tensão e Corrente (veja Fig. 8-1)

- **V_{IH}(min) — Tensão de Entrada em Nível Alto.** O nível mínimo de tensão necessário para 1 na *entrada.* Qualquer tensão abaixo deste nível não será aceita como ALTO pelo circuito lógico.
- **V_{IL}(max) — Tensão de Entrada em Nível Baixo.** O nível máximo de tensão permitido para 0 na *entrada.* Qualquer tensão acima deste nível não será aceita como BAIXO pelo circuito lógico.
- **V_{OH}(min) — Tensão de Saída em Nível Alto.** O nível mínimo de tensão na *saída* de um circuito lógico, no estado lógico 1, sob determinadas condições de carga.
- **V_{OL}(max) — Tensão de Saída em Nível Baixo.** O nível máximo de tensão na *saída* de um circuito lógico, no estado lógico 0, sob determinadas condições de carga.

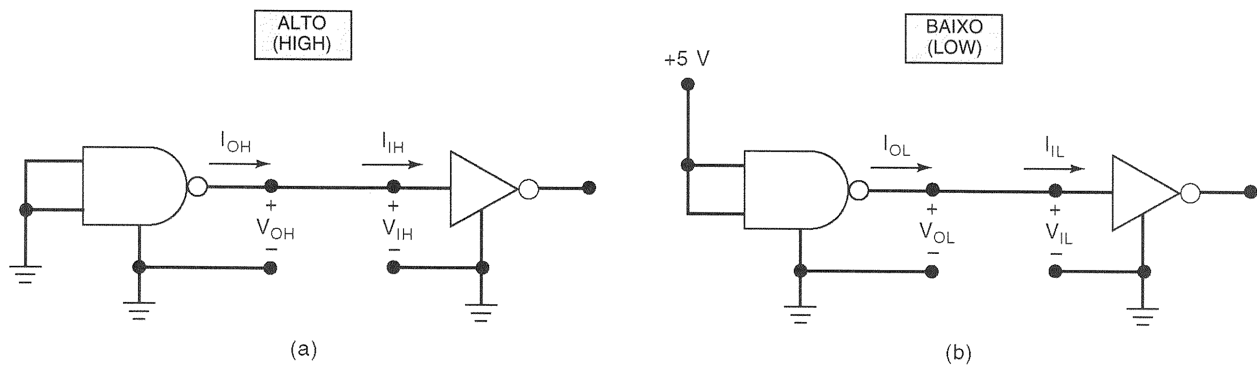


Fig. 8-1 - Tensões e correntes nos dois estados lógicos.

- I_{IH} — **Corrente de Entrada em Nível Alto.** A corrente que flui para uma entrada quando uma tensão de nível alto especificada é aplicada naquela entrada.
- I_{IL} — **Corrente de Entrada em Nível Baixo.** A corrente que flui para uma entrada quando uma tensão de nível baixo especificada é aplicada naquela entrada.
- I_{OH} — **Corrente de Saída em Nível Alto.** A corrente que flui de uma saída no estado lógico 1 sob condições de carga especificadas.
- I_{OL} — **Corrente de Saída em Nível Baixo.** A corrente que flui de uma saída no estado lógico 0 sob condições de carga especificadas.

Nota: Os sentidos reais das correntes podem ser opostos àqueles mostrados, dependendo da família lógica. Todas as descrições de fluxo de corrente neste texto se referem ao fluxo de corrente convencional (do potencial mais alto para o mais baixo). De acordo com as convenções da maioria dos manuais, a corrente que flui para dentro de um nó ou dispositivo é considerada positiva, e a corrente que sai de um nó ou dispositivo é considerada negativa.

Fan-Out

De um modo geral, uma saída de um circuito lógico necessita acionar várias entradas lógicas. O **fan-out** (também chamado *fator de carregamento*) é definido como o número *máximo* de entradas lógicas padronizadas que uma saída pode acionar confiavelmente. Por exemplo, uma porta lógica que está especificada para ter um fan-out de 10 pode acionar 10 entradas lógicas padronizadas. Se este número for excedido, as tensões dos níveis lógicos da saída não podem ser garantidas.

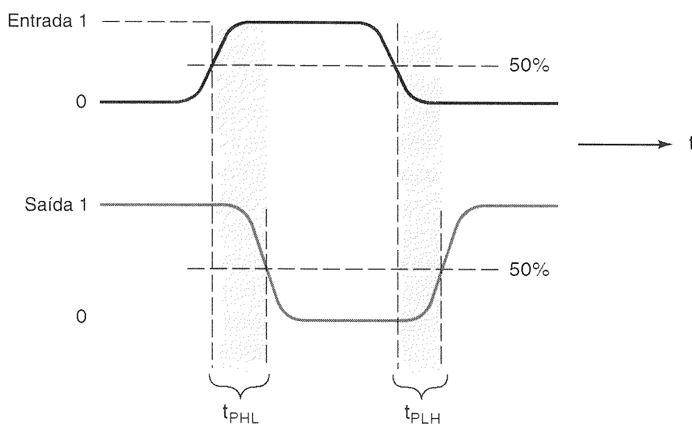


Fig. 8-2 - Atrasos de propagação.

Atrasos de Propagação

Um sinal lógico sempre sofre um atraso ao atravessar um circuito. Os dois tempos de atrasos de propagação são definidos como se segue:

- t_{PLH}. Tempo de atraso do estado 0 lógico para 1 lógico (BAIXO para ALTO, ou de LOW para HIGH).
- t_{PHL}. Tempo de atraso do estado 1 lógico para 0 lógico (ALTO para BAIXO, ou de HIGH para LOW).

A Figura 8-2 ilustra estes atrasos de propagação para um INVERSOR. Note que t_{PHL} é o atraso na saída quando ele vai de ALTO para BAIXO. Ele é medido entre os pontos que representam 50% nas transições de entrada e saída. O t_{PHL} é o atraso na saída quando ela vai de BAIXO para ALTO.

De um modo geral, t_{PHL} e t_{PLH} não têm o mesmo valor, e ambos variarão dependendo das condições de carga capacitiva. Os valores dos tempos de propagação são usados como uma medida da velocidade relativa dos circuitos lógicos. Por exemplo, um circuito lógico com valores de 10 ns é um circuito lógico mais rápido do que um com valores de 20 ns, sob determinadas condições de carga.

Requisitos de Potência

Todo CI necessita de uma certa quantidade de potência elétrica para operar. Esta potência é fornecida por uma ou mais tensões da fonte de alimentação conectadas ao(s) pino(s) de alimentação do chip. Tipicamente existe apenas um terminal para alimentação do chip, e ele é identificado como V_{CC} (para TTL) ou como V_{DD} (para dispositivos CMOS).

A quantidade de potência de que um CI necessita é determinada pela corrente, I_{CC}, que ele consome da fonte de alimentação V_{CC}, e a potência real é o produto $I_{CC} \times V_{CC}$. Para muitos CIs, a corrente consumida da fonte varia dependendo dos estados lógicos dos circuitos no chip. Por exemplo, a Fig. 8-3(a) mostra um chip NAND onde *todas* as *saídas* das portas estão em ALTO. A corrente consumida da

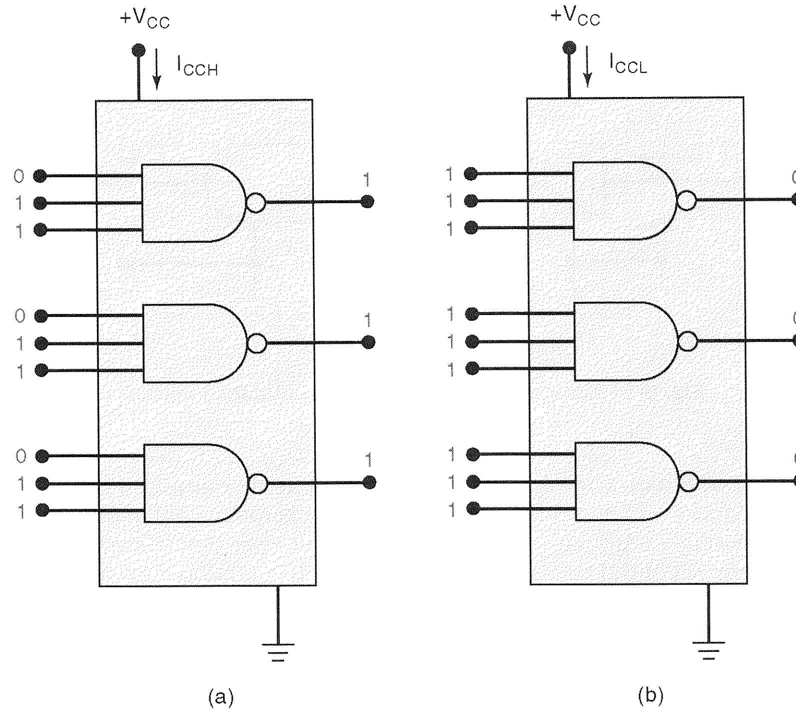


Fig. 8-3 - I_{CCH} e I_{CCL}.

fonte V_{CC} para este caso é denominada I_{CCH}. Analogamente, a Fig. 8-3(b) mostra o consumo de corrente quando *todas* as *saídas* estão em BAIXO. Esta corrente é chamada I_{CCL}. Os valores sempre são medidos com as saídas em aberto (sem carga), já que o carregamento teria um efeito sobre I_{CCH}.

Em geral, I_{CCH} e I_{CCL} têm valores diferentes. A corrente média é calculada considerando-se que as saídas das portas estarão em BAIXO durante a metade do tempo e em ALTO durante a outra metade.

$$I_{CC}(\text{med}) = \frac{I_{CCH} + I_{CCL}}{2}$$

Isto pode ser usado para calcular o consumo médio de potência como

$$P_D(\text{med}) = I_{CC}(\text{med}) \times V_{CC}$$

Produto Velocidade-Potência

As famílias de CIs digitais têm sido caracterizadas historicamente tanto pela potência quanto pela velocidade. Geralmente é mais desejável obter atrasos de propagação menores (alta velocidade) e baixos valores para dissipação de potência. Como veremos em breve, as diversas famílias e subfamílias lógicas fornecem um amplo espectro para as faixas de velocidade e potência. Uma forma comum para medir e comparar a performance total de uma família de CIs é o **produto velocidade-potência**, que é obtido multiplicando-se o atraso de propagação da porta pela dissipação de potência da mesma. Por exemplo, suponha que uma família de CIs tenha um atraso de propagação médio de 10 ns e uma dissipação média de potência de 5 mW. O produto velocidade-potência é

$$10 \text{ ns} \times 5 \text{ mW} = 50 \times 10^{-12} \text{ watt-segundo}$$
$$= 50 \text{ picojoules (pJ)}$$

Note que quando o atraso de propagação está em nanossegundos e a potência está em miliwatts, o produto velocidade-potência está em picojoules.

Obviamente, é desejável um baixo valor para o produto velocidade-potência. Os projetistas de CIs estão se esforçando continuamente para reduzir o produto velocidade-potência do produto, aumentando a velocidade do CI (isto é, reduzindo o atraso de propagação) ou diminuindo sua potência dissipada. Devido à natureza dos circuitos de chaveamento de transistores, é difícil conseguir os dois.

Imunidade ao Ruído

Campos elétricos e magnéticos podem induzir tensões nos fios de conexão entre os circuitos lógicos. Estes sinais espúrios indesejáveis são chamados de *ruído* e podem ocasionalmente fazer com que a tensão na entrada de um circuito lógico caia abaixo de $V_{IH}(\text{min})$ ou aumente além de $V_{IL}(\text{max})$, o que produziria uma operação imprevisível. A **imunidade ao ruído** de um circuito lógico se refere à capacidade do circuito em tolerar ruído sem provocar alterações espúrias na tensão de saída. Uma medida quantitativa da imunidade ao ruído é denominada **margem de ruído**, ilustrada na Fig. 8-4.

A Fig. 8-4(a) é um diagrama que mostra a faixa de tensões que pode estar presente na saída de um circuito lógico. Qualquer tensão maior do que $V_{OH}(\text{min})$ é considerada um nível lógico 1, e qualquer tensão menor do que $V_{OL}(\text{max})$ é considerada um nível lógico 0. Tensões na faixa indeterminada não deveriam aparecer na saída de um circuito lógico sob condições normais. A Fig. 8-4(b) mostra os requisitos de tensão na entrada de um circuito lógico. O circuito lógico responderá a qualquer entrada maior do que $V_{IH}(\text{min})$ como um nível lógico 1, e responderá a tensões menores

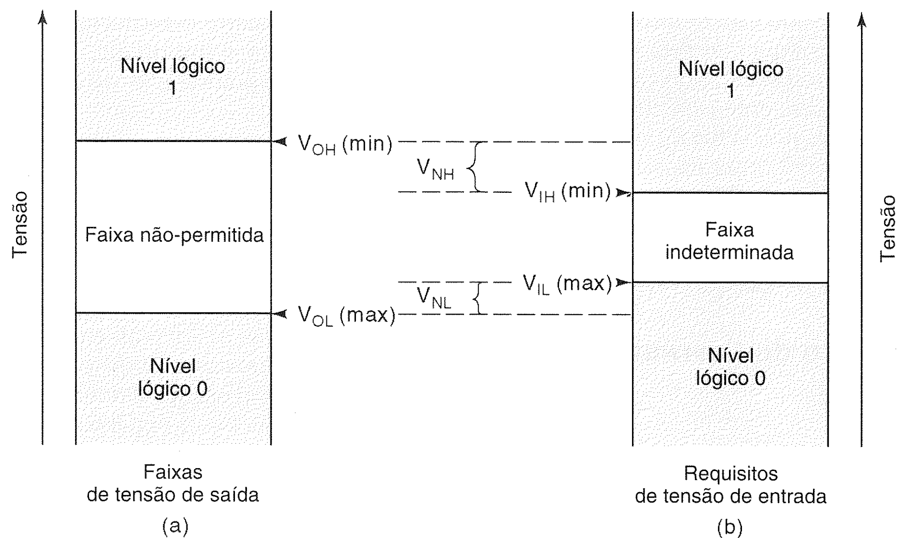


Fig. 8-4 - Margens de ruído DC.

do que V_{IL}(max) como um nível lógico 0. As tensões na faixa indeterminada produzirão uma resposta imprevisível e não deveriam ser usadas.

A *margem de ruído para estado alto* é definida como

$$V_{NH} = V_{OH}(\text{min}) - V_{IH}(\text{min}) \quad (8\text{-}1)$$

conforme está ilustrado na Fig. 8-4. V_{NH} é a diferença entre a menor saída em nível ALTO e a menor tensão de entrada necessária para um nível ALTO. Quando uma saída lógica em ALTO está acionando uma entrada de um circuito lógico, qualquer spike de ruído negativo maior do que V_{NH} que apareça na linha de sinal pode fazer com que a tensão caia na faixa indeterminada, onde uma operação imprevisível pode ocorrer.

A *margem de ruído para estado baixo* é definida como

$$V_{NL} = V_{IL}(\text{max}) - V_{OL}(\text{max}) \quad (8\text{-}2)$$

e é a diferença entre a maior saída em nível BAIXO e a maior tensão de entrada permitida para um nível BAIXO. Quando uma saída lógica em BAIXO está acionando uma entrada lógica, qualquer spike de ruído positivo maior do que V_{NL} pode fazer com que a tensão vá para a faixa indeterminada.

TABELA 8-1

Parâmetro	Min (V)	Típico (V)	Max (V)
V_{OH}	2,4	3,4	
V_{OL}		0,2	0,4
V_{IH}	2,0*		
V_{IL}			0,8*

*Normalmente apenas os valores de V_{IH} mínimo e V_{IL} máximo são fornecidos.

EXEMPLO 8-1

As especificações das tensões de entrada e saída para a família TTL padrão estão relacionadas na Tabela 8-1. Utilize estes valores para determinar o seguinte.

(a) A maior amplitude de um spike de ruído que pode ser tolerada quando uma saída em ALTO está acionando uma entrada.

(b) A maior amplitude de um spike de ruído que pode ser tolerada quando uma saída em BAIXO está acionando uma entrada.

Solução

(a) Quando uma saída está em ALTO, ela pode estar tão baixa quanto $V_{OH}(\text{min}) = 2{,}4$ V. A tensão mínima para a qual uma entrada responderá como ALTO é $V_{IH}(\text{min}) = 2{,}0$ V. Um spike negativo de ruído pode levar a tensão real abaixo dos 2,0 V se sua amplitude for maior do que

$$V_{NH} = V_{OH}(\text{min}) - V_{IH}(\text{min}) = 2{,}4\text{ V} - 2{,}0\text{ V} = 0{,}4\text{ V}$$

(b) Quando uma saída está em BAIXO, ela pode ser tão alta quanto $V_{OL}(\text{max}) = 0{,}4$ V. A tensão máxima que uma entrada responderá como nível BAIXO é $V_{IL}(\text{max}) = 0{,}8$ V. Um spike positivo de ruído pode levar a tensão real acima dos 0,8 V se sua amplitude for maior do que

$$V_{NL} = V_{IL}(\text{max}) - V_{OL}(\text{max}) = 0{,}8\text{ V} - 0{,}4\text{ V} = 0{,}4\text{ V}$$

Níveis de Tensão Inválidos

Para operar adequadamente, os níveis de tensão de entrada de um circuito lógico devem ser mantidos fora da faixa indeterminada mostrada na Fig. 8-4(b); isto é, eles devem ser menores do que V_{IL}(max) ou maiores do que V_{IH}(min). Para as especificações da série TTL padrão, apresentadas no Exemplo 8-1, isto significa que a tensão de entrada deve ser menor do que 0,8 V, ou maior do que 2,0 V. Uma tensão de entrada entre 0,8 e 2,0 V é considerada uma tensão *inválida*, que produzirá uma resposta de saída imprevisível, e, portanto, deve ser evitada. Em operação normal, uma tensão de entrada não estará dentro da região inválida, pois

ela vem de uma saída lógica que está dentro das especificações apresentadas. Entretanto, quando esta saída lógica tem problemas de funcionamento ou de sobrecarga (isto é, seu fan-out estiver sendo excedido), então sua tensão pode estar dentro da região inválida. Os níveis inválidos de tensão em um circuito digital também podem ser causados por tensões de alimentação das fontes fora da faixa aceitável. É importante saber as faixas de tensões válidas para a família lógica sendo usada, de modo que condições inválidas possam ser reconhecidas durante testes ou pesquisa de falhas.

Ação de Fornecimento de Corrente e de Absorção de Corrente

As famílias lógicas podem ser descritas de acordo com o modo como a corrente flui entre a saída de um circuito lógico e a entrada de um outro. A Fig. 8-5(a) ilustra a **ação de fornecimento de corrente**. Quando a saída da porta 1 está em ALTO, ela fornece uma corrente I_{IH} para a entrada da porta 2, que funciona essencialmente como uma resistência para a terra. Assim, a saída da porta 1 funciona como uma *fornecedora* de corrente para a entrada da porta 2. Podemos pensar nisto como uma torneira que opera como uma *fonte* de água.

A **ação de absorção de corrente** está ilustrada na Fig. 8-5(b). Neste caso, o circuito de entrada da porta 2 está representado como uma resistência ligada a $+V_{CC}$, o terminal positivo da fonte de alimentação. Quando a saída da porta 1 vai para o estado BAIXO, a corrente fluirá no sentido mostrado, do circuito de entrada da porta 2, através da resistência de saída da porta 1, para a terra. Em outras palavras, no estado BAIXO o circuito de saída que aciona a entrada da porta 2 deve ser capaz de *absorver* a corrente, I_{IL}, vinda daquela entrada. Podemos pensar nisto como um *ralo* pelo qual a água está fluindo.

A distinção entre o fornecimento de corrente e a absorção de corrente é importante, e se tornará mais aparente conforme examinarmos as diversas famílias lógicas.

Encapsulamentos de CIs

Os desenvolvimentos e avanços nos circuitos integrados continuam cada vez mais velozes. O mesmo pode ser dito quanto aos encapsulamentos de CIs. Existe uma variedade de tipos de encapsulamentos que diferem no tamanho físico, nas condições ambientais e de consumo de energia sobre as quais o dispositivo pode operar confiavelmente, e no modo pelo qual o encapsulamento do CI é montado na placa de circuito impresso. A Fig. 8-6 mostra três encapsulamentos de CIs representativos.

O encapsulamento na Fig. 8-6(a) é o **DIP** (*dual-in-line package*), que já existe por um bom tempo. Seus pinos (ou *leads*) estão dispostos nos dois lados maiores do encapsulamento retangular. O dispositivo mostrado é um DIP de 24 pinos. Note a presença do chanfro num dos lados, que é usado para localizar o pino 1. Alguns DIPs utilizam um pequeno ponto na superfície superior do encapsulamento para localizar o pino 1. Os pinos que saem do encapsulamento DIP estão dispostos de tal forma que o CI pode ser colocado num soquete, ou inserido nos furos de uma placa de circuito impresso. O espaçamento entre os pinos (**passo entre pinos**) tipicamente é de 100 mils (um mil é um milésimo de polegada). Os encapsulamentos DIP ainda são os encapsulamentos mais populares para a construção de protótipos e para experiências educacionais.

Quase todas as novas placas eletrônicas que são produzidas utilizando equipamentos automáticos de fabricação não usam mais os encapsulamentos DIP, cujos pinos são inseridos através dos furos na placa de circuito impresso. Os novos métodos de fabricação utilizam a **tecnologia de montagem**

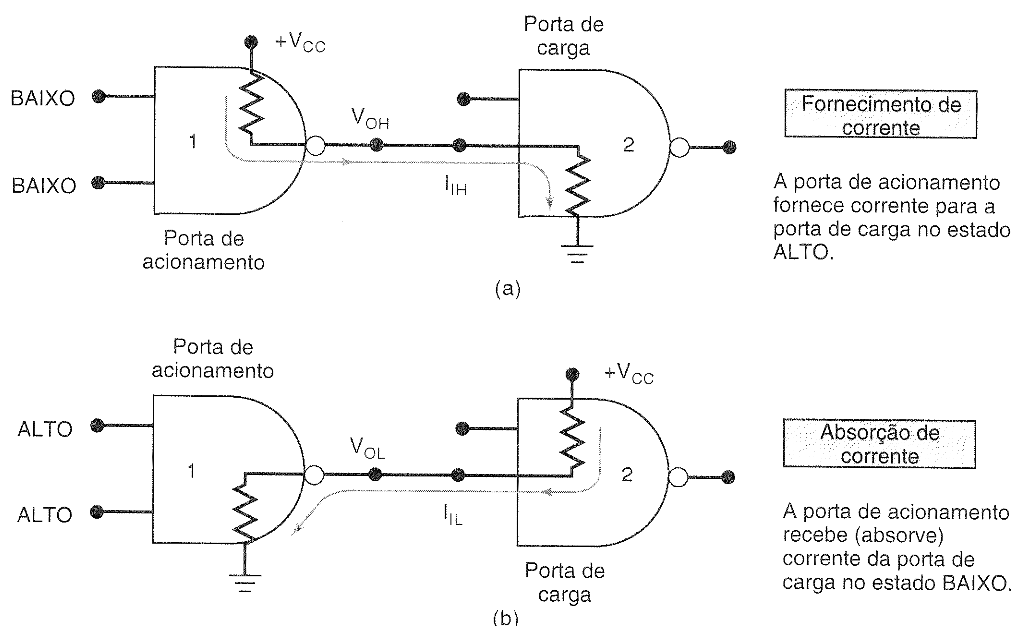


Fig. 8-5 - Comparação entre as ações de fornecimento de corrente e de absorção de corrente.

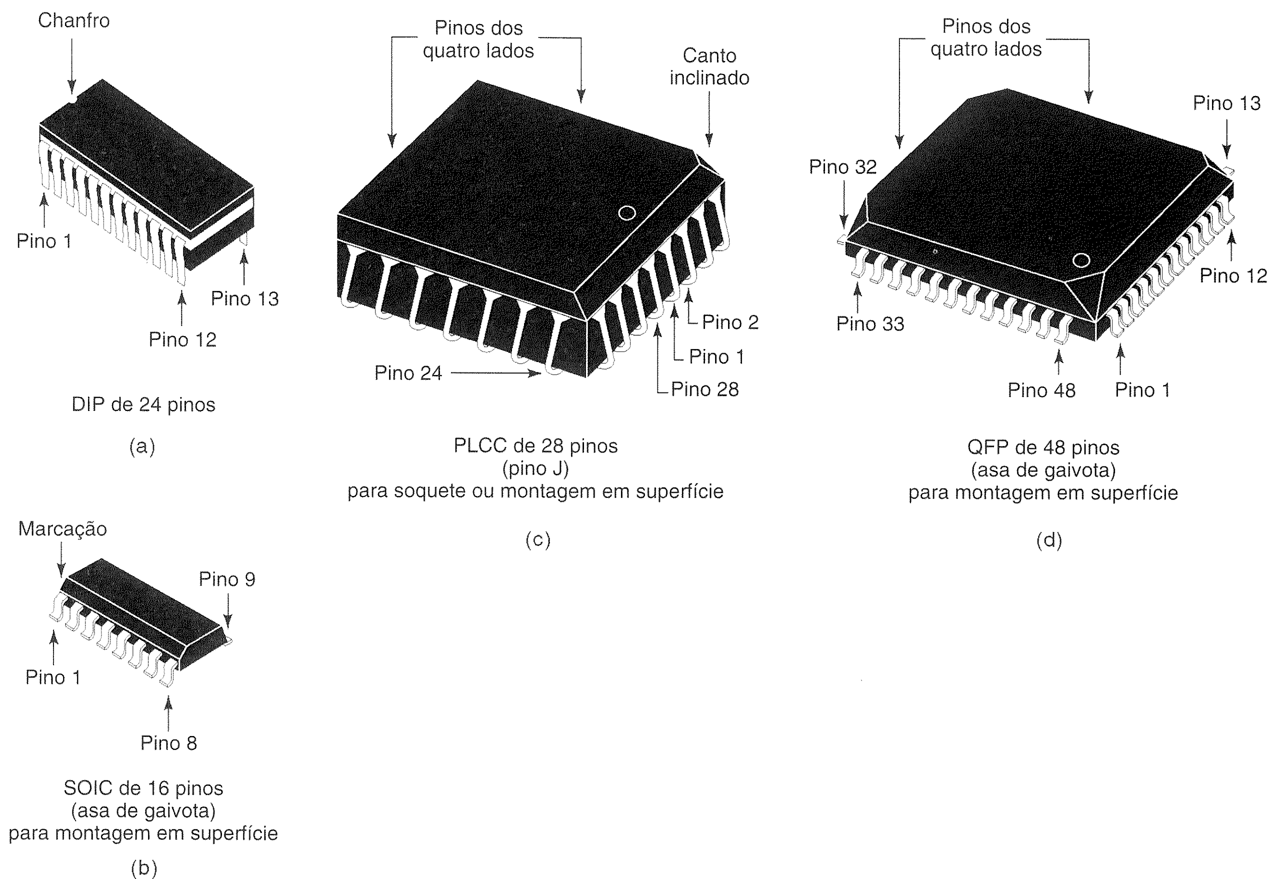


Fig. 8-6 - Encapsulamentos comuns de CIs.

em superfície, que coloca um CI sobre contatos elétricos na superfície da placa. Eles são mantidos no lugar por uma pasta de solda, e a placa inteira é aquecida para realizar as conexões de solda. A precisão da máquina de colocação permite um espaçamento entre pinos bem pequeno. Os pinos nos encapsulamentos para montagem em superfície são dobrados, apresentando uma área adequada para a solda. A forma destes pinos resultou no apelido de encapsulamento "*gull-wing*" ("asa de gaivota"). Muitos encapsulamentos diferentes estão disponíveis para dispositivos de montagem em superfície. Alguns dos encapsulamentos mais comuns usados para CIs lógicos são mostrados na Fig. 8-6. A Tabela 8-2 fornece a definição de cada sigla com as respectivas dimensões.

A necessidade para mais e mais conexões em CIs complexos resultou em outro encapsulamento muito popular, que tem pinos em todos os quatro lados do chip. O PLCC tem pinos no formato da letra J, que se curvam sob o CI conforme mostrado na Fig. 8-6(c). Estes dispositivos podem ser montados diretamente em placas de circuito impresso, mas também podem ser colocados em soquetes especiais. Isto é comumente usado, para componentes que possivelmente precisarão ser substituídos em reparos ou atualizados, como dispositivos lógico-programáveis ou unidades centrais de processamento em computadores. Os encapsulamentos para montagem em superfície QFP e TQFP possuem pinos *gull-wing* nos quatro lados, conforme mostrado na Fig. 8-6(d).

TABELA 8-2 Encapsulamentos de CIs.

Sigla	Nome do Encapsulamento	Altura	Passo entre Pinos
DIP	Dual-In-line Package	200 mils (5,1 mm)	100 mils (2,54 mm)
SOIC	Small Outline Integrated Circuit	2,65 mm	50 mils (1,27 mm)
SSOP	Shrink Small Outline Package	2,0 mm	0,65 mm
TSSOP	Thin Shrink Small Outline Package	1,1 mm	0,65 mm
TVSOP	Thin Very Small Outline Package	1,2 mm	0,4 mm
PLCC	Plastic Leaded Chip Carrier	4,5 mm	1,27 mm
QFP	Quad Flat Pack	4,5 mm	0,635 mm
TQFP	Thin Quad Flat Pack	1,6 mm	0,5 mm

Questões de Revisão

1. Defina cada um dos seguintes termos: V_{OH}, V_{IL}, I_{OL}, I_{IH}, t_{PLH}, t_{PHL}, I_{CCL} e I_{CCH}.
2. *Verdadeiro ou falso*: Se um circuito lógico tem um fan-out de 5, o circuito tem cinco saídas.
3. *Verdadeiro ou falso*: A margem de ruído em nível ALTO é a diferença entre V_{IH}(min) e V_{CC}.
4. *Verdadeiro ou falso*: Uma família lógica com t_{pd}(med) = 12 ns e P_D(med) = 15 mW tem um produto velocidade-potência maior do que uma com 8 ns e 30 mW.
5. Descreva a diferença entre fornecimento de corrente e absorção de corrente.
6. Que tipos de encapsulamentos de CIs podem ser colocados em soquetes?
7. Que encapsulamento tem pinos dobrados sob o CI?
8. Como os encapsulamentos para montagem em superfície diferem dos DIPs?
9. Um dispositivo TTL padrão funcionará com um nível de entrada de 1,7 V?

8-2 A FAMÍLIA LÓGICA TTL

Durante a preparação deste livro, CIs de pequena e média escalas de integração (SSI e MSI) ainda se encontravam disponíveis na tecnologia da série **TTL** padrão, que está disponível há 30 anos. Esta série original de dispositivos, e seus descendentes na família TTL, teve uma enorme influência nas características de todos os dispositivos lógicos atuais. Dispositivos TTL ainda são utilizados como lógica auxiliar que conecta os dispositivos mais complexos em sistemas digitais. Eles também são usados como circuitos de interface para dispositivos que necessitam de acionamento com corrente mais alta. Embora a família bipolar TTL como um todo esteja em declínio, iniciaremos nossa apresentação sobre CIs lógicos com os dispositivos que moldaram a tecnologia digital.

O circuito lógico básico TTL é a porta NAND. Seu diagrama de circuito detalhado, mostrado na Fig. 8-7(a), tem diversas características específicas. Primeiramente, note que o transistor Q_1 tem dois emissores; logo, tem duas junções base-emissor (B-E) que podem ser usadas para fazer Q_1 conduzir. Este transistor de entrada de *múltiplos emissores* pode ter até oito emissores para uma porta NAND de oito entradas.

Repare também que na saída do circuito os transistores Q_3 e Q_4 estão num arranjo denominado **totem-pole**. Como veremos em breve, em operação normal, ou Q_3 ou Q_4 estará conduzindo, dependendo do estado lógico da saída.

Operação do Circuito — Estado BAIXO

Embora este circuito pareça extremamente complexo, podemos simplificar sua análise utilizando o equivalente a diodo do transistor de múltiplos emissores, Q_1, conforme mostrado na Fig. 8-7(b). Os diodos D_2 e D_3 representam as duas junções B-E de Q_1, e D_4 é a junção base-coletor (B-C). Na análise a seguir, utilizaremos esta representação para Q_1.

Primeiramente, vamos considerar o caso em que a saída está em BAIXO. A Fig. 8-8(a) mostra esta situação com as entradas A e B em + 5 V. A tensão de + 5 V nos catodos de D_2 e D_3 os deixa cortados, e eles praticamente não conduzirão corrente alguma. A fonte de + 5 V fornecerá corrente através de R_1 e D_4 para a base de Q_2, que conduz. A corrente do emissor de Q_2 fluirá para a base de Q_4 e o faz conduzir. Ao mesmo tempo, o fluxo de corrente no coletor de Q_2 produz uma queda de tensão sobre R_2, que reduz a tensão no coletor de Q_2 para um valor que é insuficiente para fazer Q_3 conduzir.

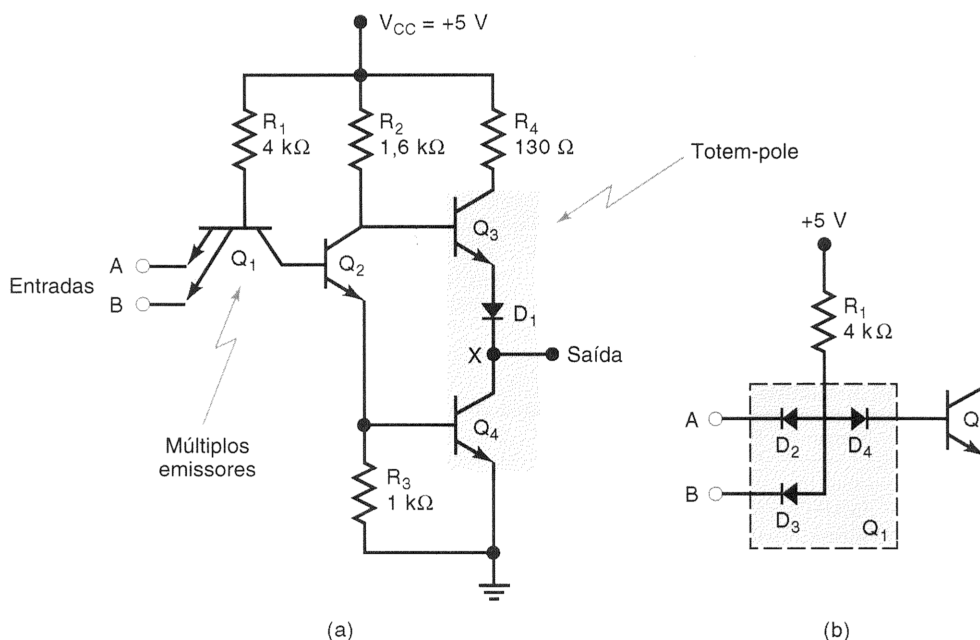


Fig. 8-7 - (a) Porta NAND básica TTL; (b) equivalente a diodo para Q_1.

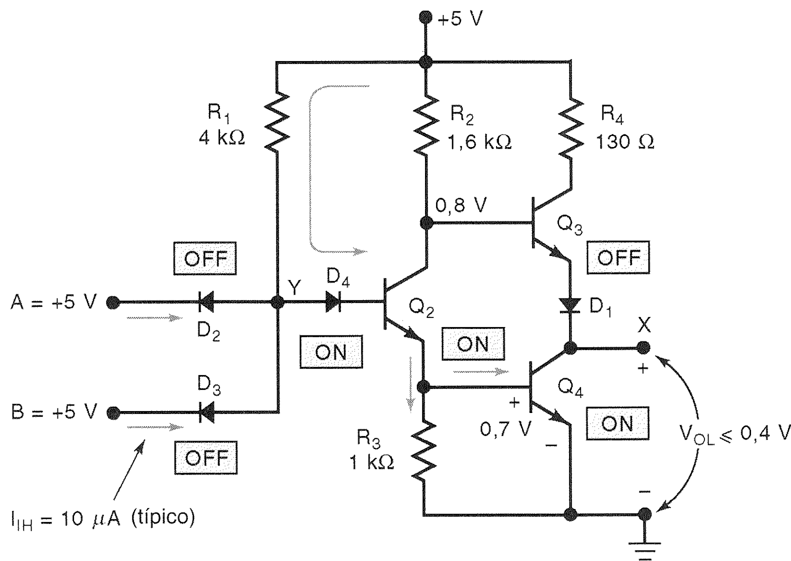
A tensão no coletor de Q_2 é de aproximadamente 0,8 V. Isto é porque o emissor de Q_2 está a 0,7 V em relação à terra, devido à tensão direta entre B-E de Q_4, e o coletor de Q_2 está a 0,1 V em relação ao seu emissor devido ao V_{CE}(sat). Este valor de 0,8 V na base de Q_3 não é suficiente para polarizar diretamente a junção B-E de Q_3 e o diodo D_1. Na verdade, D_1 é necessário para manter Q_3 cortado nesta situação.

Com Q_4 conduzindo, o terminal de saída, *X*, estará com uma tensão muito baixa, já que a resistência de Q_4, quando conduz, será baixa (1 a 25 Ω). Na verdade, a tensão de saída, V_{OL}, dependerá de quanta corrente de coletor Q_4 conduz. Com Q_3 cortado, não existe corrente vindo do terminal da fonte de + 5 V, através de R_4. Como veremos, a corrente do coletor de Q_4 virá das entradas TTL às quais o terminal *X* estiver conectado.

É importante notar que as entradas em ALTO, *A* e *B*, terão que fornecer apenas a pequena corrente de fuga dos diodos. Tipicamente, esta corrente, I_{IH}, é por volta de 10 μA para a temperatura ambiente.

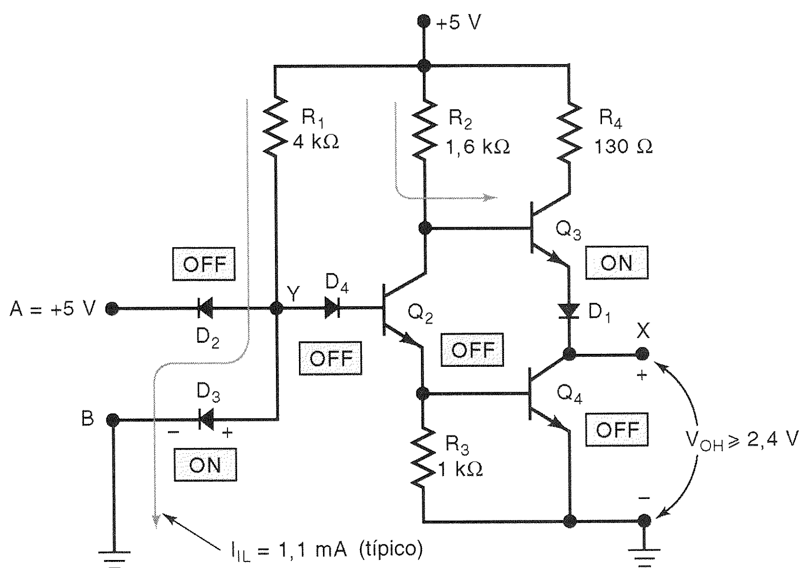
Operação do Circuito — Estado ALTO

A Fig. 8-8(b) mostra a situação em que a saída do circuito está em ALTO. Esta situação pode ser produzida conectando-se uma entrada ou ambas as entradas em BAIXO. Aqui, a entrada *B* está conectada na terra. Isto vai polarizar D_3 dire-



Condições de entrada	Condições de saída
A e B estão ambas em ALTO (≥ 2 V)	Q_3 OFF
Correntes de entrada são muito baixas I_{IH} = 10 μA	Q_4 ON logo V_X é BAIXO (≤ 0,4 V)

(a) Saída em BAIXO



Condições de entrada	Condições de saída
A ou B ou ambos em BAIXO (≤ 0,8 V)	Q_4 OFF
A corrente flui para a terra através do terminal de entrada em BAIXO. I_{IL} = 1,1 mA	Q_3 atua como seguidor de emissor e V_{OH} ≥ 2,4 V, geralmente 3,6 V

(b) Saída em ALTO

Fig. 8-8 - Porta NAND TTL nos seus dois estados de saída.

tamente, de modo que a corrente fluirá do terminal de + 5 V, através de R_1 e D_3, pelo terminal B para a terra. A tensão direta sobre D_3 manterá o ponto Y em aproximadamente 0,7 V. Esta tensão não é suficiente para polarizar diretamente D_4 e a junção B-E de Q_2 para condução.

Com Q_2 cortado, não existe corrente de base para Q_4, e ele corta. Como não existe corrente de coletor em Q_2, a tensão na base de Q_3 será grande o suficiente para polarizar Q_3 e D_1, de modo que Q_3 conduz. Na verdade, Q_3 opera como um seguidor de emissor, porque essencialmente o terminal de saída X está no seu emissor. Sem carga conectada do ponto X para a terra, V_{OH} estará em torno de 3,4 a 3,8 V, pois duas quedas de diodo de 0,7 V (B-E de Q_3 e D_1) devem ser subtraídas dos 5 V aplicados à base de Q_3. Esta tensão diminuirá com a carga, porque a carga receberá corrente do emissor de Q_3, que por sua vez recebe corrente de base através de R_2, aumentando portanto a queda de tensão sobre R_2.

É importante notar que existe uma corrente substancial fluindo através do terminal de entrada B para a terra. Esta corrente, I_{IL}, é tipicamente de 1,1 mA. A entrada B em BAIXO funciona como um *absorvedor* para esta corrente para a terra.

Ação de Absorção de Corrente

Uma saída TTL atua como um absorvedor de corrente no estado BAIXO, pois ela *recebe* corrente da entrada da porta que está acionando. A Fig. 8-9 mostra uma porta TTL acionando a entrada de uma outra porta (a carga) para ambos os estados de tensão de saída. Para a situação de estado de saída em BAIXO apresentada na Fig. 8-9(a), o transistor Q_4 da porta de acionamento está conduzindo (ON) e essencialmente "liga" o ponto X na terra. Esta tensão em nível BAIXO, em X, polariza diretamente a junção B-E de Q_1, e a corrente flui, como mostrado, de volta através de Q_4. Assim, Q_4 está realizando uma ação de absorção da corrente proveniente da entrada (I_{IL}) da porta de carga. Freqüentemente nos referiremos a Q_4 como o **transistor de absorção de corrente** ou como o **transistor de pull-down,** porque ele leva a tensão de saída para seu estado BAIXO.

Ação de Fornecimento de Corrente

Uma saída TTL atua como uma fornecedora de corrente no estado ALTO. Isto é mostrado na Fig. 8-9(b), onde o transistor Q_3 está fornecendo a corrente de entrada, I_{IH}, necessária para o transistor Q_1 da porta de carga. Conforme mencionado anteriormente, esta corrente é uma pequena corrente de fuga de polarização reversa (tipicamente 10 μA). Freqüentemente nos referiremos a Q_3 como o **transistor de fornecimento de corrente** ou como o **transistor de pull-up**.

Circuito de Saída Totem-Pole

Diversos pontos devem ser comentados em relação ao arranjo totem-pole do circuito de saída TTL, como mostrado na Fig. 8-9, já que não é claro por que ele é usado. A mesma lógica poderia ser obtida eliminando-se Q_3 e D_1 e conectando-se a parte de baixo de R_4 no coletor de Q_4. Mas isto significaria que Q_4 conduziria uma corrente bem maior no seu estado de saturação (5 V/130 $\Omega \approx$ 40 mA). Com Q_3 no circuito, não existirá corrente através de R_4 no estado de saída BAIXO. Isto é importante pois mantém baixa a dissipação de potência do circuito.

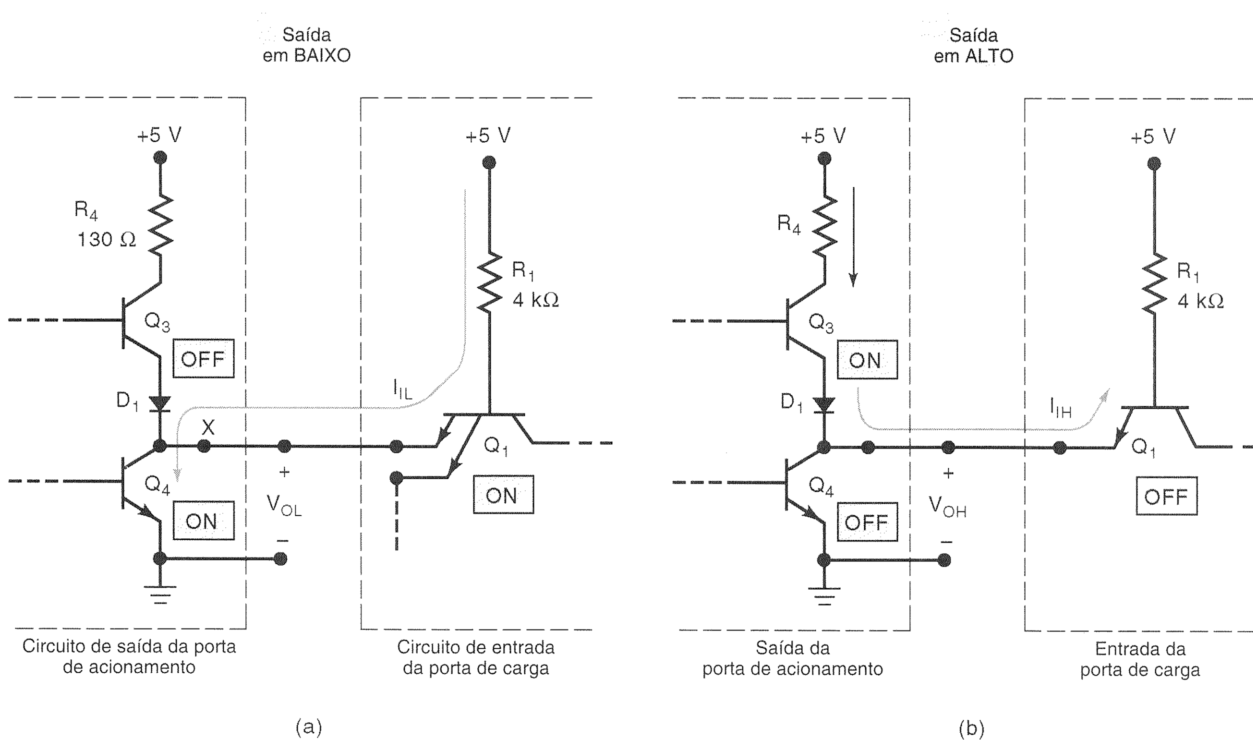


Fig. 8-9 - (a) Quando a saída TTL está no estado BAIXO, Q_4 atua como absorvedor de corrente, recebendo sua corrente da carga. (b) No estado de saída ALTO, Q_3 atua como fornecedor de corrente, fornecendo corrente para a porta de carga.

Uma outra vantagem deste arranjo ocorre no estado de saída em ALTO. Neste caso, Q_3 atua como um seguidor de emissor, com sua baixa impedância de saída (tipicamente 10 Ω). Esta baixa impedância de saída acarreta uma pequena constante de tempo para carregar qualquer carga capacitiva na saída. Esta ação (comumente chamada *pull-up ativo*) proporciona tempos de subida muito curtos para as formas de onda nas saídas TTL.

Uma desvantagem do arranjo de saída totem-pole ocorre durante a transição de BAIXO para ALTO. Infelizmente, Q_4 pára de conduzir mais lentamente do que Q_3 passa a conduzir, e portanto existe um período de poucos nanossegundos durante o qual ambos os transistores estão conduzindo, e uma corrente relativamente grande (30 a 40 mA) será consumida da fonte de 5 V. Isto pode representar um problema que examinaremos mais tarde.

Porta NOR TTL

A Fig. 8-10 mostra o circuito interno para uma porta NOR TTL. Não faremos uma análise detalhada deste circuito, mas é importante notar como ele se compara com o circuito NAND da Fig. 8-8. Na entrada, podemos ver que o circuito NOR *não utiliza um transistor com múltiplos emissores*; em vez disto, cada entrada é aplicada ao emissor de um transistor em separado. Na saída, o circuito NOR utiliza o mesmo arranjo totem-pole como o circuito NAND.

Resumo

Todos os circuitos TTL têm uma estrutura similar. As portas NAND e AND utilizam transistores de múltiplos emissores nas entradas; as portas NOR e OR usam transistores de entrada separados. Em qualquer dos casos, a entrada será o catodo (região N) de uma junção P-N, de modo que uma tensão de entrada em ALTO manterá a junção reversamente polarizada, e apenas uma pequena corrente de fuga (I_{IH}) fluirá. Por outro lado, uma tensão de entrada em BAIXO faz a junção conduzir, e uma corrente relativamente grande (I_{IL}) fluirá de volta para a fonte do sinal. A maioria, mas não todos, dos circuitos TTL tem algum tipo de configuração de saída totem-pole. Existem algumas exceções que discutiremos mais tarde.

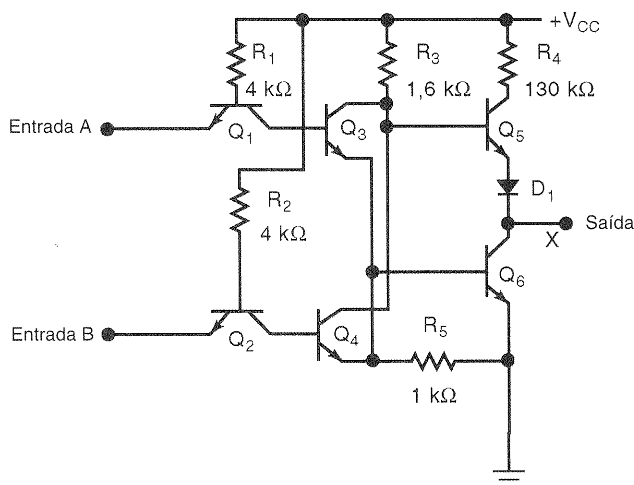


Fig. 8-10 - Circuito da porta NOR TTL.

Questões de Revisão

1. *Verdadeiro ou falso:* Uma saída TTL atua como um absorvedor de corrente no estado BAIXO.
2. Em qual estado de entrada TTL flui a maior quantidade de corrente?
3. Relacione as vantagens e desvantagens de uma saída totem-pole.
4. Qual transistor TTL é o transistor de pull-up no circuito NAND?
5. Qual transistor TTL é o transistor de pull-down no circuito NOR?
6. Como o circuito TTL NOR difere do circuito NAND?

8-3 CARACTERÍSTICAS DA SÉRIE TTL PADRÃO

Em 1964, a Texas Instruments Corporation introduziu a primeira linha de CIs TTL padrão. A série 54/74, como é chamada, tem sido uma das famílias lógicas de CIs mais amplamente utilizada. Vamos nos referir a ela simplesmente como série 74, tendo em vista que a principal diferença entre as versões 54 e 74 é que os dispositivos da série 54 podem operar em faixas de temperatura e de tensão de alimentação maiores. Muitos fabricantes de semicondutores produzem CIs TTL. Felizmente, todos eles usam o mesmo sistema de numeração, de modo que o número básico do CI é o mesmo de um fabricante para o outro. Cada fabricante, entretanto, usualmente acrescenta seu próprio prefixo especial ao número do CI. Por exemplo, a Texas Instruments usa o prefixo SN, a National Semiconductor utiliza DM, e a Signetics usa S. Assim, dependendo do fabricante, você pode ver um chip de porta NOR quádruplo identificado como DM7402, SN7402, S7402 ou alguma outra designação similar. A parte importante é o número 7402, que é o mesmo para todos os fabricantes.

Como aprendemos no Cap. 4, existem diversas séries de dispositivos lógicos na família TTL (74, 74LS, 74S etc). A série padrão original e suas descendentes imediatas (74, 74LS, 74S) não são mais recomendadas pelos fabricantes para utilização em novos projetos. Apesar disso, ainda existe no mercado demanda suficiente para mantê-los em produção. A compreensão das características que definem as capacidades e limitações de qualquer dispositivo lógico é vital. Esta seção definirá estas características utilizando a série padrão TTL como exemplo, pois ela é a "vovó de todas elas". Mais adiante introduziremos as outras séries TTL e compararemos suas características com as da série padrão.

Folha de Características do Fabricante

Para ilustrar as características da série TTL padrão, utilizamos o CI 7400 de quatro portas NAND. Podemos encontrar todas as informações de que necessitamos sobre um CI

consultando o manual do fabricante para a família do CI. A Fig. 8-11 é a folha de características (*data sheet*) do CI de portas NAND 5400/7400, mostrando as condições de operação recomendadas, as características elétricas e as características de chaveamento. A maioria dos parâmetros discutidos nos parágrafos seguintes a esta seção pode ser achada nesta folha de características. À medida que discutirmos cada parâmetro, você deve consultar esta folha de características para ver de onde veio a informação.

Faixas de Tensão de Alimentação e de Temperatura

Tanto a série 74 quanto a série 54 usam uma tensão de alimentação nominal (V_{CC}) de 5 V. A série 74 operará de modo confiável na faixa de 4,75 a 5,25 V, enquanto a série 54 pode tolerar uma variação na fonte de 4,5 a 5,5 V. A série 74 é projetada para operar apropriadamente em temperaturas ambientes variando de 0 a 70°C, enquanto a série 54 pode tolerar de −55 a +125°C. Por causa de sua tolerância maior para variações de temperatura e tensão, a série 54 é mais cara. Ela é empregada apenas em aplicações onde uma operação confiável deve ser mantida sobre uma extensa faixa de condições. Exemplos são as aplicações militares e espaciais.

Níveis de Tensão

Os níveis de tensão lógicos de entrada e saída para a série TTL padrão podem ser encontrados na folha de características da Fig. 8-11. A Tabela 8-3 relaciona estes níveis. Os valores máximos e mínimos mostrados são para condições de pior caso da fonte de alimentação, da temperatura e de condições de carga. Uma inspeção da tabela revela um nível lógico 0 de saída garantido de V_{OL} = 0,4 V, que é 400 mV menor do que a tensão de nível 0 necessária na entrada, V_{IL} = 0,8 V. Isto significa que a margem de ruído DC garantida no estado 0 é 400 mV. Isto é,

$$V_{NL} = V_{IL}(\text{max}) - V_{OL}(\text{max}) = 0{,}8\ \text{V} - 0{,}4\ \text{V}$$
$$= 0{,}4\ \text{V} = 400\ \text{mV}$$

Analogamente, a saída em nível lógico 1, V_{OH}, tem um mínimo garantido de 2,4 V, que é 400 mV maior do que a tensão de nível lógico 1 necessária na entrada, V_{IH} = 2,0 V. Assim, a margem de ruído DC para o estado ALTO é de 400 mV.

Recommended operating conditions

		SN5400			SN7400			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			−0.4			−0.4	mA
I_{OL}	Low-level output current			16			16	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

Electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS†			SN5400			SN7400			UNIT
				MIN	TYP‡	MAX	MIN	TYP‡	MAX	
V_{IK}	V_{CC} = MIN,	I_I = −12 mA				−1.5			−1.5	V
V_{OH}	V_{CC} = MIN,	V_{IL} = 0.8 V,	I_{OH} = −0.4 mA	2.4	3.4		2.4	3.4		V
V_{OL}	V_{CC} = MIN,	V_{IH} = 2 V,	I_{OL} = 16 mA		0.2	0.4		0.2	0.4	V
I_I	V_{CC} = MAX,	V_I = 5.5 V				1			1	mA
I_{IH}	V_{CC} = MAX,	V_I = 2.4 V				40			40	μA
I_{IL}	V_{CC} = MAX,	V_I = 0.4 V				−1.6			−1.6	mA
I_{OS}§	V_{CC} = MAX			−20		−55	−18		−55	mA
I_{CCH}	V_{CC} = MAX,	V_I = 0 V			4	8		4	8	mA
I_{CCL}	V_{CC} = MAX,	V_I = 4.5 V			12	22		12	22	mA

† For conditions shown as MIN or MAX, use the appropriate value specified under recommended operating conditions.
‡ All typical values are at V_{CC} = 5 V, T_A = 25°C.
§ Not more than one output should be shorted at a time.

Switching characteristics, V_{CC} = 5 V, TA = 25°C (see note 2)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	TEST CONDITIONS		MIN	TYP	MAX	UNIT
t_{PLH}	A or B	Y	R_L = 400 Ω	C_L = 15 pF		11	22	ns
t_{PHL}						7	15	ns

NOTE 2: See General Information Section for load circuits and voltage waveforms.

Fig. 8-11 - Folha de características para o CI de portas NAND 7400. (Cortesia da Texas Instruments.)

$$V_{NH} = V_{OH}(\text{min}) - V_{IH}(\text{min}) = 2{,}4\ \text{V} - 2{,}0\ \text{V}$$
$$= 0{,}4\ \text{V} = 400\ \text{mV}$$

Assim, as margens de ruído DC *garantidas para pior caso* para a série 74 são ambas de 400 mV.

Faixas Máximas de Tensão

Os valores de tensão na Tabela 8-3 *não incluem* os limites absolutos máximos, além dos quais a vida útil do CI pode ser prejudicada. As tensões aplicadas em qualquer entrada da série de CIs 74 padrão nunca devem exceder + 5,5 V. Uma tensão maior do que + 5,5 V aplicada a um emissor de entrada pode causar problemas na junção B-E de Q_1.

Também existe um limite na tensão *negativa* máxima que pode ser aplicada em uma entrada TTL. Este limite, − 0,5 V, é causado pelo fato de a maioria dos circuitos TTL empregar diodos de proteção em cada entrada. Estes diodos foram deixados de lado de nossa análise anterior de propósito, pois eles não têm função na operação normal do circuito. Eles são conectados de cada entrada para a terra, para limitar as excursões de tensão negativa nas entradas que freqüentemente ocorrem quando sinais lógicos apresentam oscilações após mudanças de estado. Com estes diodos, não podemos aplicar mais do que − 0,5 V numa entrada, pois os diodos de proteção começariam a conduzir e consumiriam uma corrente substancial, provavelmente fazendo com que o diodo entrasse em curto, resultando em uma entrada permanentemente defeituosa.

Dissipação de Potência

Uma porta NAND TTL padrão consome uma potência média de 10 mW. Isto é um resultado de I_{CCH} = 4 mA e I_{CCL} = 12 mA, que produz I_{CC}(med) = 8 mA e P_D(med) = 8 mA × 5 V = 40 mW. Este 40 mW é a potência total necessária para todas as quatro portas do chip. Logo, uma porta NAND requer uma potência média de 10 mW.

Atrasos de Propagação

A porta AND TTL padrão tem atrasos de propagação típicos de t_{PLH} = 11 ns e t_{PHL} = 7 ns, o que resulta num atraso de propagação *médio* t_{pd}(med) de 9 ns.

Fan-Out

Uma saída TTL padrão pode tipicamente acionar 10 entradas TTL padrão. Algumas folhas de características de CIs fornecem explicitamente o fan-out do dispositivo, mas este não é o caso para a folha de características da Fig. 8-11. Mais adiante, veremos como o fan-out pode ser determinado dos valores de corrente para entrada e saída apresentados na Fig. 8-11.

TABELA 8-3 Níveis de tensão da série 74 padrão.

	Mínimo	Típico	Máximo
V_{OL}	—	0,2	0,4
V_{OH}	2,4	3,4	—
V_{IL}	—	—	0,8
V_{IH}	2,0	—	—

EXEMPLO 8-2

Consulte a folha de características para o CI NAND quádruplo de duas entradas 7400, na Fig. 8-11. Determine a *máxima* dissipação de potência média e o *máximo* atraso de propagação médio de uma *única* porta.

Solução

Procure nas características elétricas pelos valores *máximos* de I_{CCH} e I_{CCL}. Os valores são 8 mA e 22 mA, respectivamente. O I_{CC} médio é portanto (8 + 22)/2 = 15 mA. A potência média é obtida multiplicando-se por V_{CC}. A folha de características indica que esses valores de I_{CC} foram obtidos quando V_{CC} estava com seu valor máximo (5,25 V para a série 74). Logo, temos

$$P_D(\text{med}) = 15\ \text{mA} \times 5{,}25\ \text{V} = 78{,}75\ \text{mW}$$

para a potência consumida pelo CI *completo*. Podemos determinar a potência de uma porta NAND dividindo isto por 4:

$$P_D(\text{med}) = 19{,}7\ \text{mW por porta}$$

Tendo em vista que esta potência média foi calculada usando os valores máximos de tensão e corrente, ela é a potência média máxima que uma porta NAND do 7400 consumirá sob condições de pior caso. Projetistas freqüentemente utilizam os valores de pior caso para garantir que seus circuitos funcionarão sob todas as condições.

Os atrasos de propagação máximos para uma porta NAND 7400 estão relacionados como

$$t_{PLH} = 22\ \text{ns} \qquad t_{PHL} = 15\ \text{ns}$$

portanto o atraso de propagação máximo é

$$t_{pd}(\text{med}) = \frac{22 + 15}{2} = 18{,}5\ \text{ns}$$

Novamente, este é o atraso de propagação médio máximo possível para pior caso.

8-4 SÉRIES TTL APERFEIÇOADAS

A série TTL padrão de CIs oferece uma grande variedade de portas, FFs e monoestáveis em integração de pequena escala (SSI), e contadores, registradores, codificadores/decodificadores, circuitos aritméticos e muitas outras funções lógicas na sua linha de integração em média escala (MSI). Os circuitos básicos da série TTL padrão formam a parte central de diversas outras séries TTL; mas os dispositivos TTL padrão (série 74) raramente são usados em novos projetos devido à melhor performance das novas séries TTL que têm sido desenvolvidas ao longo dos anos. Estas outras séries TTL — freqüentemente denominadas "subfamílias" — fornecem uma ampla faixa de capacidades de velocidade e potência.

Séries 74L e 74H

As séries 74L e 74H foram desenvolvidas para obter versões de TTL com baixa potência e alta velocidade, respectivamente. Ambas têm o mesmo circuito básico da série 74 padrão, mas diferenças nos valores dos componentes dos circuitos dão a estas duas séries suas características específicas. Comparada com a série 74, a série 74L era uma versão de baixa potência que consumia menos (1 mW) mas com um atraso de propagação muito maior (33 ns); a série 74H era uma versão de alta velocidade que tinha um atraso de propagação reduzido (6 ns), mas com um consumo de potência mais alto (23 mW). Nenhuma destas séries está em produção atualmente, pois suas performances foram superadas pelas novas séries.

TTL Schottky, Série 74S

As séries 74, 74H e 74L operam utilizando chaveamento com saturação, no qual muitos transistores, quando conduzindo, estão na condição de saturação. Esta operação causa um atraso de tempo de armazenamento, t_S, quando os transistores comutam do estado de condução (ON) para o estado de não-condução (OFF), e ele limita a velocidade de chaveamento do circuito.

A série 74S reduz este atraso do tempo de armazenamento não permitindo que o transistor fique muito saturado. Ela consegue isto utilizando diodos Schottky conectados entre a base e o coletor de cada transistor, como mostra a Fig. 8-12(a). O diodo Schottky tem uma tensão direta de apenas 0,25 V. Assim, quando a junção B-C se torna polarizada diretamente no limite da saturação, o diodo conduzirá e desviará uma parte da corrente de entrada para fora da base. Isto reduz o excesso de corrente de base e diminui o atraso do tempo de armazenamento na ida para o corte.

Como ilustrado na Fig. 8-12(a), é dado um símbolo especial para a combinação transistor/diodo. Este símbolo é usado para todos os transistores no diagrama do circuito para a porta NAND 74S00 mostrada na Fig. 8-12(b). Esta porta NAND 74S00 tem um atraso de propagação médio de apenas 3 ns, que é duas vezes menor do que o atraso do 74H00. Note a presença dos diodos D_1 e D_2 para limitar as tensões de entrada negativas.

Os circuitos da série 74S também usam valores de resistores menores para ajudar a melhorar os tempos de chaveamento. Isto aumenta a dissipação média de potência do circuito para aproximadamente 20 mW, praticamente a mesma que a da 74H. Os circuitos 74S também usam um par Darlington (Q_3 e Q_4) para obterem um tempo de subida na saída mais curto quando transicionarem de ON para OFF.

TTL Schottky de Baixa Potência, Série 74LS (LS-TTL)

A série 74LS é uma versão de menor potência e menor velocidade da série 74S. Ela utiliza a combinação transistor/diodo Schottky, mas com valores de resistores mais altos do que os da série 74S. Os resistores de maior valor reduzem os requisitos de potência do circuito, mas provocam um aumento nos tempos de chaveamento.

Uma porta NAND na série 74LS terá, tipicamente, um atraso de propagação médio de 9,5 ns e uma dissipação

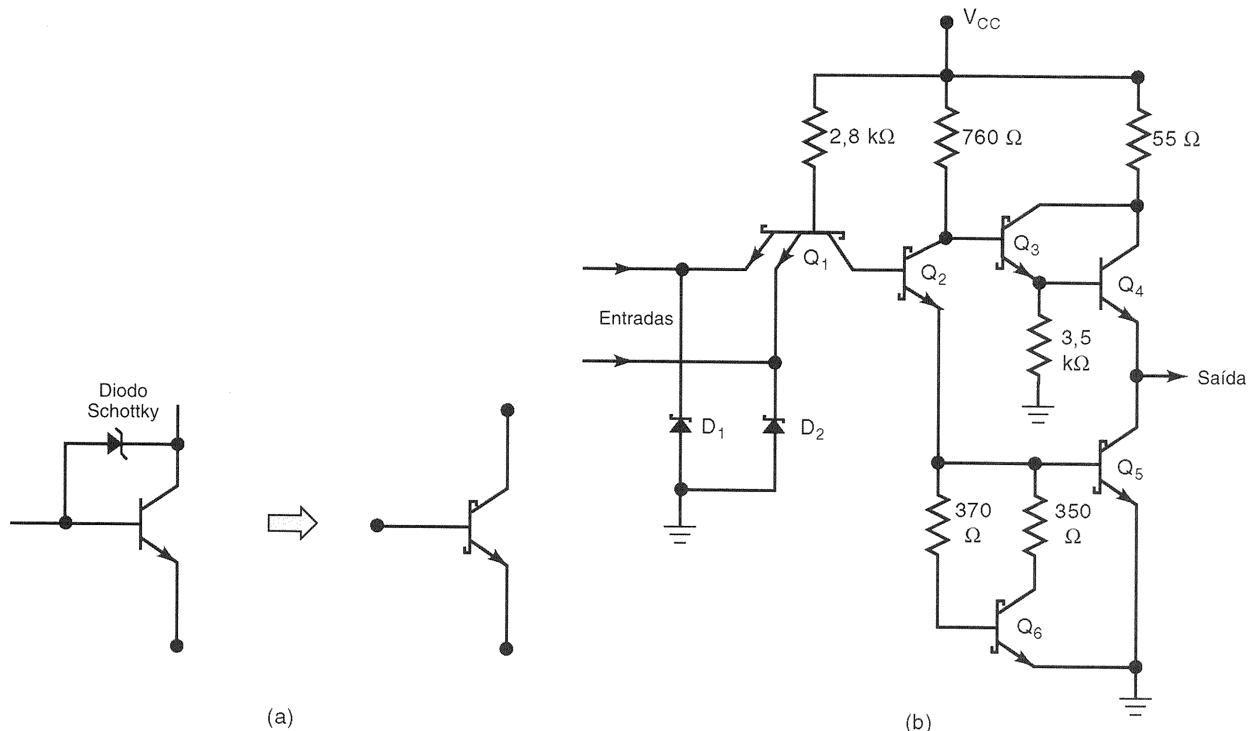


Fig. 8-12 - (a) Combinação transistor e diodo Schottky; (b) porta NAND básica na série S-TTL. (Cortesia da Fairchild, uma companhia da Schlumberger.)

média de potência de 2 mW. Como ela tem quase a mesma velocidade da série TTL padrão com um requisito de potência bem menor, a série 74LS se tornou a base de sustentação da família TTL nos anos 1980. Entretanto, sua posição de liderança tem sido tomada pela família CMOS, particularmente pelas séries 74HC e 74HCT.

TTL Schottky Avançada, Série 74AS (AS-TTL)

Inovações no projeto de circuitos integrados levaram ao desenvolvimento de duas séries TTL avançadas: Schottky avançada (74AS) e Schottky avançada de baixa potência (74ALS). A série 74AS fornece uma considerável melhoria na velocidade em relação à série 74S, com requisitos de potência bem menores. A comparação é mostrada na Tabela 8-4 para uma porta NAND de cada série. Esta comparação mostra claramente as vantagens da série 74AS. Ela é a série TTL mais rápida, e seu produto velocidade-potência é significativamente mais baixo do que o da 74S. A 74AS tem outras melhorias, incluindo valores menores para correntes de entrada (I_{IL}, I_{IH}), o que resulta em um maior fan-out do que a série 74S.

TABELA 8-4

	74S	74AS
Atraso de propagação	3 ns	1,7 ns
Dissipação de potência	20 mW	8 mW
Produto velocidade-potência	60 pJ	13,6 pJ

TTL Schottky Avançada de Baixa Potência, Série 74ALS

Esta série oferece uma melhoria sobre a série 74LS tanto na velocidade quanto na dissipação de potência, como os números da Tabela 8-5 ilustram. A série 74ALS tem o produto velocidade-potência mais baixo e a menor dissipação de potência de todas as séries TTL. Seu custo ligeiramente mais alto tem evitado que ela substitua a 74LS, a não ser em aplicações de alta velocidade.

TABELA 8-5

	74LS	74ALS
Atraso de propagação	9,5 ns	4ns
Dissipação de potência	2 mW	1,2 mW
Produto velocidade-potência	19 pJ	4,8 pJ

TTL Fast-74F

Esta é a série TTL mais nova. Ela utiliza uma nova técnica para fabricação de circuitos integrados que reduz as capacitâncias entre os dispositivos internos para alcançar atrasos de propagação reduzidos. Uma porta NAND típica tem um atraso de propagação médio de 3 ns e um consumo de potência de 6 mW. Os CIs nesta série são designados com a letra F no seu número. Por exemplo, o 74F04 é um chip com seis inversores. Essas séries TTL de alta performance (74AS, 74ALS, 74F) servem a pequenos nichos da indústria digital.

Comparação das Características das Séries TTL

A Tabela 8-6 apresenta os valores típicos para algumas das mais importantes características de cada uma das séries TTL. Todos os dados de performance, exceto para a máxima taxa de clock, são para uma porta NAND de cada série. A taxa de clock máxima é especificada como a freqüência máxima que pode ser usada para comutar um flip-flop J-K. Isto dá uma medida útil da faixa de freqüência na qual cada série de CIs pode ser operada.

EXEMPLO 8-3

Utilize a Tabela 8-6 para calcular as margens de ruído DC para um CI 74LS típico. Como se comparam com as margens de ruído obtidas para TTL padrão na Seção 8-3?

Solução

$$V_{NH} = V_{OH}(\text{min}) - V_{IH}(\text{min}) = 2{,}7\ \text{V} - 2{,}0\ \text{V} = 0{,}7\ \text{V}$$

TABELA 8-6 Características típicas das séries TTL.

	74	74S	74LS	74AS	74ALS	74F
Índices de performance						
Atraso de propagação (ns)	9	3	9,5	1,7	4	3
Dissipação de potência (mW)	10	20	2	8	1,2	6
Produto velocidade-potência (pJ)	90	60	19	13,6	4,8	18
Taxa máxima de clock (MHz)	35	125	45	200	70	100
Fan-out (mesma série)	10	20	20	40	20	33
Parâmetros de tensão						
V_{OH}(min)	2,4	2,7	2,7	2,5	2,5	2,5
V_{OL}(max)	0,4	0,5	0,5	0,5	0,4	0,5
V_{IH}(min)	2,0	2,0	2,0	2,0	2,0	2,0
V_{IL}(max)	0,8	0,8	0,8	0,8	0,8	0,8

comparada com V_{NH} = 0,4 V para TTL padrão.

$$V_{NL} = V_{IL}(\max) - V_{OL}(\max) = 0{,}8\ V - 0{,}5\ V = 0{,}3\ V$$

comparada com V_{NL} = 0,4 V para TTL.

EXEMPLO 8-4

Qual das séries TTL pode acionar o maior número de dispositivos de entrada da mesma série?

Solução

A série 74AS tem o maior fan-out (40). Isto significa que uma porta NAND de um 74AS00 pode acionar 40 entradas padronizadas de outros dispositivos 74AS. Se quisermos determinar o número de entradas de uma série TTL *diferente* que uma saída pode acionar, precisaremos saber as correntes de entrada e saída das duas séries. Trataremos disso em detalhes na próxima seção.

Questões de Revisão

1. **(a)** Qual das séries TTL é a melhor em altas freqüências?
   (b) Qual das séries TTL tem a maior margem de ruído em nível ALTO?
   (c) Quais séries se tornaram essencialmente obsoletas?
   (d) Quais séries utilizam um diodo especial para reduzir o tempo de chaveamento?
   (e) Qual série seria melhor para um circuito alimentado por baterias funcionando a 10 MHz?
2. Admitindo o mesmo custo para ambos, por que você escolheria usar um contador 74ALS193 em vez de um 74LS193 ou um 74AS193, em um circuito operando com um clock de 40 MHz?
3. Identifique os transistores de pull-up e pull-down para o circuito 74S na Fig. 8-12.

8-5 FAN-OUT E CARREGAMENTO PARA TTL

É importante compreender o que determina o fan-out ou a capacidade de acionamento da saída de um CI. A Fig. 8-13(a) mostra uma saída TTL padrão no estado BAIXO conectada para acionar diversas entradas TTL padrão. O transistor Q_4 está conduzindo (ON) e está absorvendo uma quantidade de corrente I_{OL}, que é a soma das correntes I_{IL} de cada entrada. No seu estado ON, a resistência de coletor para emissor de Q_4 é muito pequena, mas não é zero, e portanto a corrente I_{OL} produzirá uma queda de tensão V_{OL}. Esta tensão não deve exceder o limite V_{OL}(max) do CI. Isto limita o valor máximo de I_{OL} e o número de cargas que podem ser acionadas.

Para ilustrar, suponha que os CIs sejam da série 74, e que cada I_{IL} é de 1,6 mA. Da Tabela 8-6, vemos que a série 74 tem V_{OL}(max) = 0,4 V e V_{IL}(max) = 0,8 V. Vamos supor também que Q_4 pode absorver até 16 mA antes que a tensão de saída alcance V_{OL}(max) = 0,4 V. Isto significa que ele pode absorver a corrente de até 16 mA/1,6 mA = 10 cargas. Se ele for conectado a mais de 10 cargas, seu I_{OL} aumentará e provocará um aumento de V_{OL} para um valor acima de 0,4 V. Isto geralmente é indesejável porque reduz a margem de ruído nas entradas do CI [lembre-se, $V_{NL} = V_{IL}(\max) - V_{OL}(\max)$]. Na verdade, se V_{OL} ultrapassa V_{IL}(max) = 0,8 V, ela estará na faixa indeterminada.

Uma situação parecida ocorre no estado ALTO e está ilustrada na Fig. 8-13(b). Q_3 está atuando como um seguidor de emissor que está fornecendo uma corrente total I_{OH} que

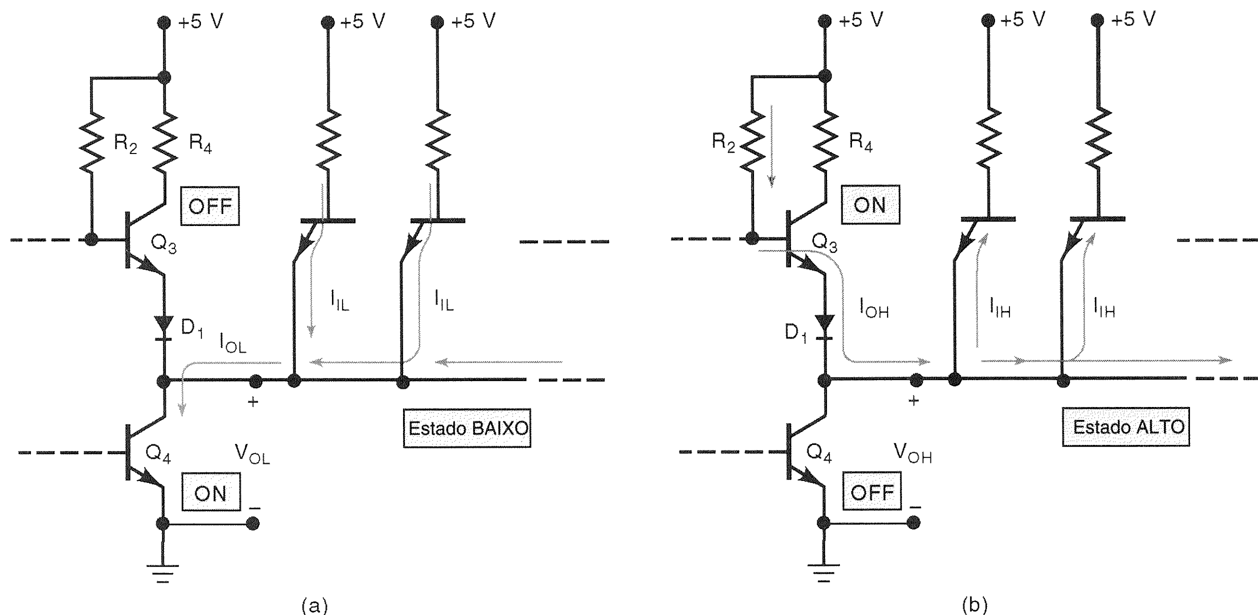


Fig. 8-13 - Correntes quando uma saída TTL está acionando diversas entradas.

é a soma das correntes I_{IH} das diferentes entradas TTL. Se cargas em demasia estiverem sendo acionadas, esta corrente I_{OH} se tornará suficientemente grande para causar quedas de tensão em R_2, na junção base-emissor de Q_3, e em D_1, de modo a levar V_{OH} abaixo de V_{OH}(min). Isto também é indesejável, já que reduz a margem de ruído no estado ALTO e poderia até mesmo deixar V_{OH} na faixa indeterminada.

Em resumo, a saída TTL tem um limite, I_{OL}(max), da quantidade de corrente que pode absorver no estado BAIXO. Ela também tem um limite, I_{OH}(max), da quantidade de corrente que pode fornecer no estado ALTO. Estes limites de corrente de saída não devem ser excedidos se os níveis de tensão de saída precisarem ficar dentro das faixas especificadas.

Determinando o Fan-Out

Para determinar quantas entradas diferentes a saída de um CI pode acionar, você precisa saber a capacidade de corrente da saída [isto é, I_{OL}(max) e I_{OH}(max)] e os requisitos de corrente de cada entrada (isto é, I_{IL} e I_{IH}). Esta informação sempre está presente de algum modo na folha de características do fabricante do CI. Os exemplos a seguir ilustrarão várias situações.

EXEMPLO 8-5

Quantas entradas de portas NAND 7400 podem ser acionadas pela saída de uma porta NAND 7400?

Solução

Vamos considerar primeiramente o estado BAIXO, conforme ilustra a Fig. 8-14. Consulte a folha de características na Fig. 8-11 e encontre

$$I_{OL}(\text{max}) = 16 \text{ mA}$$
$$I_{IL}(\text{max}) = 1{,}6 \text{ mA}$$

Isto informa que uma saída do 7400 pode absorver no máximo 16 mA e que cada entrada do 7400 fornecerá no máximo 1,6 mA de volta para a saída da porta acionadora. Assim, o número de entradas que podem ser acionadas no estado BAIXO é

$$\text{fan-out (BAIXO)} = \frac{I_{OL}(\text{max})}{I_{IL}(\text{max})} = \frac{16 \text{ mA}}{1{,}6 \text{ mA}} = 10$$

(*Nota:* A informação para I_{IL} é na verdade $-1{,}6$ mA. O sinal de menos é usado para indicar que esta corrente flui para *fora* do terminal de entrada; podemos ignorar este sinal aqui para nossos propósitos.) O estado ALTO é analisado da mesma maneira. Consulte a folha de características para encontrar os valores de I_{OH} e I_{IH}, ignorando qualquer sinal negativo.

$$I_{OH}(\text{max}) = 0{,}4 \text{ mA} = 400\ \mu\text{A}$$
$$I_{IH}(\text{max}) = 40\ \mu\text{A}$$

Logo, o número de entradas que podem ser acionadas no estado ALTO é

$$\text{fan-out (ALTO)} = \frac{I_{OH}(\text{max})}{I_{IH}(\text{max})} = \frac{400\ \mu\text{A}}{40\ \mu\text{A}} = 10$$

Estes resultados indicam que o fan-out é 10 em ambos os estados. Assim, a porta NAND 7400 pode acionar até outras 10 portas NAND 7400. Se o fan-out (BAIXO) e o fan-out (ALTO) não são iguais, como acontece às vezes, o fan-out é escolhido como o menor dos dois.

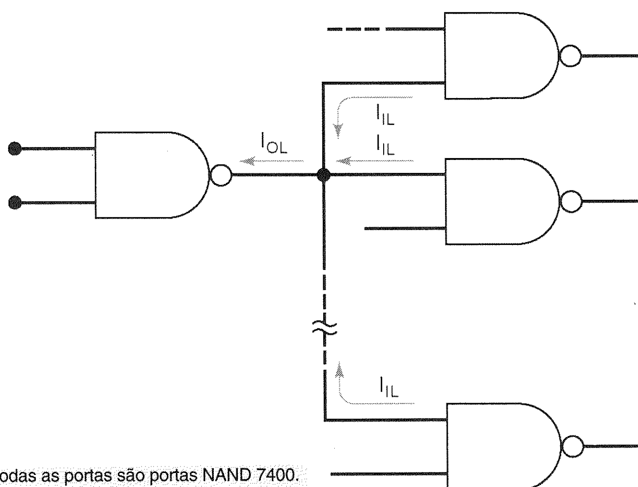


Fig. 8-14 - Exemplo 8-5.

EXEMPLO 8-6

Consulte a folha de características no Apêndice e determine quantas portas NAND 74ALS20 podem ser acionadas pela saída de uma outra 74ALS20.

Solução

A folha de características fornece os seguintes valores:

$$I_{OH}(\text{max}) = 0{,}4 \text{ mA} = 400\ \mu\text{A}$$
$$I_{OL}(\text{max}) = 8 \text{ mA}$$
$$I_{IH}(\text{max}) = 20\ \mu\text{A}$$
$$I_{IL}(\text{max}) = 0{,}1 \text{ mA}$$

Considerando o estado ALTO primeiro, temos

$$\text{fan-out (ALTO)} = \frac{400\ \mu\text{A}}{20\ \mu\text{A}} = 20$$

Para o estado BAIXO temos

$$\text{fan-out (BAIXO)} = \frac{8 \text{ mA}}{0{,}1 \text{ mA}} = 80$$

Neste caso, o fan-out global é escolhido como 20, pois é o menor dos dois valores. Logo, um 74ALS20 pode acionar outras 20 entradas 74ALS20.

Em equipamentos mais antigos, você notará que a maioria dos CIs freqüentemente era escolhida da mesma família lógica. Nos sistemas digitais atuais, o mais provável é aparecer uma combinação de várias famílias lógicas. Conseqüentemente, os cálculos de carregamento e fan-out não são tão diretos como eram antes. Um bom método para a determinação do carregamento de qualquer saída digital é o seguinte:

Passo 1. Some I_{IH} de todas as entradas conectadas em uma saída. Esta soma deve ser menor do que a especificação do I_{OH} da saída.

Passo 2. Some I_{IL} de todas as entradas conectadas em uma saída. Esta soma deve ser menor do que a especificação do I_{OL} da saída.

A Tabela 8-7 mostra as especificações limites para as correntes de entrada e saída de portas lógicas simples de várias famílias TTL. Repare que alguns valores de corrente são números negativos. Esta convenção é usada para mostrar o sentido do fluxo da corrente. Valores positivos indicam a corrente fluindo para dentro do ponto especificado, seja uma entrada ou uma saída. Valores negativos indicam que a corrente flui para fora do ponto especificado. Conseqüentemente, todos os valores de I_{OH} são negativos, uma vez que a corrente flui para fora do pino de saída (fornece corrente), e todos os valores de I_{OL} são positivos, pois a corrente de carga flui para dentro do pino de saída em direção a terra (absorve corrente). Analogamente, I_{IH} é positivo, enquanto I_{IL} é negativo. Quando se calculam carregamento e fan-out, conforme descrito anteriormente, você pode ignorar estes sinais.

EXEMPLO 8-7

A saída de uma porta NAND 74ALS00 está acionando três entradas de portas 74S e uma entrada 7406. Determine se existe um problema de carregamento.

TABELA 8-7 Parâmetros de corrente para portas lógicas das séries TTL.*

	Saídas		Entradas	
Série TTL	I_{OH}	I_{OL}	I_{m}	I_{IL}
74	−0,4 mA	16 mA	40 μA	−1,6 mA
74S	−1 mA	20 mA	50 μA	−2 mA
74LS	−0,4 mA	8 mA	20 μA	−0,4 mA
74AS	−2mA	20 mA	20 μA	−0,5 mA
74ALS	−0,4 mA	8 mA	20 μA	−0,1 mA
74F	−1 mA	20 mA	20 μA	−0,6 mA

*Alguns dispositivos podem ter parâmetros diferentes de corrente de entrada e saída. Consulte sempre a folha de características.

Solução

1. Some todos os valores de I_{IH}:

$$3 \cdot (I_{IH} \text{ para 74S}) + 1 \cdot (I_{IH} \text{ para 74})$$
$$\text{Total} = 3 \cdot (50\ \mu A) + 1 \cdot (40\ \mu A) = 190\ \mu A$$

O I_{OH} para a saída 74ALS é de 400 μA (max), que é maior do que a soma das cargas (190 μA). Isto não representa nenhum problema quando a saída está em ALTO.

2. Some todos os valores de I_{IL}:

$$3 \cdot (I_{IL} \text{ para 74S}) + 1 \cdot (I_{IL} \text{ para 74})$$
$$\text{Total} = 3 \cdot (2\ mA) + 1 \cdot (1{,}6\ mA) = 7{,}6\ mA$$

O I_{OL} para a saída 74ALS é de 8 mA (max), que é maior do que a soma das cargas (7,6 mA). Isto não representa nenhum problema quando a saída está em BAIXO.

EXEMPLO 8-8

A saída da porta NAND 74ALS00 no Exemplo 8-7 precisa ser usada para acionar algumas entradas 74ALS, além das cargas descritas no Exemplo 8-7. Quantas entradas 74ALS adicionais poderiam ser acionadas pela saída sem provocar sobrecarga?

Solução

Dos cálculos do Exemplo 8-7, somente no estado BAIXO estamos perto de uma sobrecarga. Uma entrada 74ALS tem um I_{IL} de 0,1 mA. A corrente máxima de absorção (I_{OL}) é 8 mA, e a corrente de carga é 7,6 mA (conforme o cálculo do Exemplo 8-7). A corrente adicional que a saída pode absorver é calculada por

$$\text{corrente adicional} = I_{OLmax} - \text{soma das cargas } (I_{IL})$$
$$= 8\ mA - 7{,}6\ mA = 0{,}4\ mA$$

Esta saída pode acionar mais quatro entradas 74ALS que tenham I_{IL} de 0,1 mA.

EXEMPLO 8-9

A saída de um inversor 74AS04 está fornecendo o sinal de CLEAR para um registrador paralelo construído a partir de flip-flops 74AS74. Qual é o maior número de entradas *CLR* destes FFs que esta porta pode acionar?

Solução

As especificações de entrada para as entradas de flip-flops não são sempre as mesmas que as das entradas de portas da mesma família. Consulte a folha de características do 74AS74 no Apêndice. As entradas *D* e de clock são similares às entradas das portas na Tabela 8-7. Entretanto, as entradas *PRE* e *CLR* têm especificações para I_{IH} = 40 μA e I_{IL} = 1,8 mA. O 74AS04 tem especificações para I_{OH} = 2 mA e I_{OL} = 20 mA.

Número máximo de entradas (ALTO) = 2 mA/40 μA = 50
Número máximo de entradas (BAIXO) = 20 mA/1,8 mA
= 11,11

Devemos limitar o fan-out a 11 entradas *CLR*.

Questões de Revisão

1. Que fatores determinam I_{OL}(max) de um dispositivo?
2. Quantas entradas 7407 podem ser acionadas por um chip 74AS?
3. O que pode acontecer se uma saída TTL for ligada em mais entradas do que está preparada para acionar?
4. Quantas entradas $\overline{CP}$ de um 74S112 podem ser acionadas por uma saída 74LS04? E por uma saída 74F00?

8-6 OUTRAS CARACTERÍSTICAS TTL

Muitas outras características da lógica TTL devem ser entendidas se se quiser usá-la inteligentemente numa aplicação de sistema digital.

Entradas Desconectadas (Flutuando)

Qualquer entrada para um circuito TTL que é deixada desconectada (aberta) atua exatamente como o nível lógico 1 aplicado àquela entrada, pois em ambos os casos a junção base-emissor ou o diodo na entrada não estará diretamente polarizado. Isto significa que em *qualquer* CI TTL, *todas* as entradas são 1s se não estiverem conectadas a algum sinal lógico ou na terra. Quando uma entrada está desconectada, diz-se que está **flutuando**.

Entradas Não-Usadas

Freqüentemente, nem todas as entradas de um CI TTL estão sendo usadas em determinada aplicação. Um exemplo comum é quando nem todas as entradas de uma porta lógica são necessárias para a função lógica requisitada. Por exemplo, suponha que precisamos da operação lógica $\overline{AB}$ e estamos usando um chip que tem uma porta NAND de três entradas. Os modos possíveis de tratar isto estão mostrados na Fig. 8-15.

Na Fig. 8-15(a), a entrada não-utilizada está desconectada, o que significa que atua como nível lógico 1. A saída da porta NAND é portanto $x = \overline{A \cdot B \cdot 1} = \overline{A \cdot B}$, que é o resultado desejado. Embora a lógica esteja correta, é indesejável deixar uma entrada desconectada, pois ela atuará como uma antena, que é susceptível a captar sinais irradiados que fariam com que a porta não operasse adequadamente. Uma técnica melhor é mostrada na Fig. 8-15(b). Aqui a entrada não-utilizada é conectada a + 5 V através de um resistor de 1 kΩ, de modo que o nível lógico é 1. O resistor de 1 kΩ serve simplesmente para proteção de corrente das junções base-emissor das entradas da porta no caso de spikes na fonte de alimentação. Esta mesma técnica pode ser usada para portas AND, já que 1 em uma entrada não-usada não afetará a saída. Até 30 entradas não-utilizadas podem compartilhar o mesmo resistor de 1 kΩ ligado a V_{CC}.

Uma terceira possibilidade é mostrada na Fig. 8-15(c), onde a entrada não-utilizada é ligada em uma entrada usada. Isto é satisfatório contanto que o circuito acionador da entrada *B* não fique com seu fan-out excedido. Esta técnica pode ser usada para *qualquer* tipo de porta. Para portas OR e NOR, as entradas não-usadas não podem ficar desconectadas nem ligadas a + 5 V, pois isto produziria um nível lógico constante na saída (1 para OR, 0 para NOR) independentemente das outras entradas. Em vez disso, para estas portas as entradas não-usadas devem ser conectadas na terra (0 V) para nível 0, ou devem ser ligadas em entradas usadas, como na Fig. 8-15(c).

Conectando Entradas Juntas

Quando duas (ou mais) entradas TTL na mesma porta são conectadas juntas para formar uma entrada comum, como na Fig. 8-15(c), geralmente esta entrada comum representará uma carga que é a soma das correntes de carga de cada entrada individual. A única exceção é para portas NAND e AND. Para estas portas, a carga da entrada em estado BAIXO *será a mesma de uma entrada única*, não importando quantas entradas estão conectadas juntas.

Para ilustrar, considere que cada entrada da porta NAND de três entradas na Fig. 8-15(c) tem 0,5 mA para I_{IL} e 20 μA para I_{IH}. A entrada comum *B* representará, portanto, uma carga de entrada de 40 μA no estado ALTO, mas de apenas 0,5 mA no estado BAIXO. O mesmo seria válido para uma porta AND. Se fosse uma porta OR ou uma NOR, a entrada *B* comum representaria uma carga de entrada de 40 μA no estado ALTO e 1 mA no estado BAIXO.

A razão para esta característica pode ser entendida verificando-se o diagrama de circuito da porta NAND TTL da Fig. 8-8(b). A corrente I_{IL} está limitada pela resistência R_1. Mesmo que as entradas *A* e *B* fossem ligadas juntas e aterradas, esta corrente não se alteraria; ela apenas se dividiria e fluiria por caminhos paralelos através dos diodos D_2 e D_3. A situação é diferente para portas OR e NOR, já que elas não utilizam transistores com múltiplos emissores, mas têm transistores de entrada separados para cada entrada, conforme vimos na Fig. 8-10.

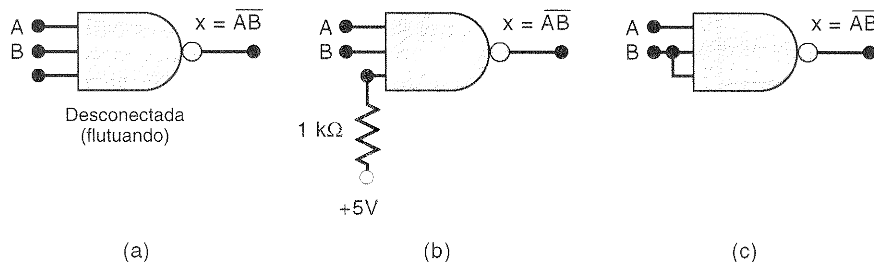
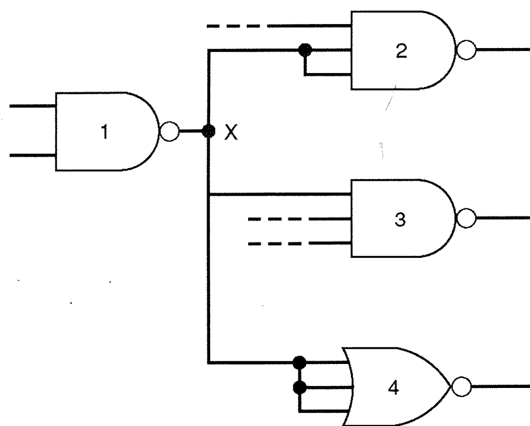


Fig. 8-15 - Três modos de tratar entradas lógicas não-usadas.



Carregamento na saída da porta 1			
ALTO		BAIXO	
Corrente de Carga	Porta	Corrente de Carga	Porta
40 μA	2	0,4 mA	2
20 μA	3	0,4 mA	3
60 μA	4	1,2 mA	4
120 μA	Total	2,0 mA	Total

Fig. 8-16 - Exemplo 8-10.

EXEMPLO 8-10

Determine a carga que a saída X está acionando na Fig. 8-16. Suponha que cada porta é um dispositivo da série 74LS com $I_{IH} = 20\ \mu A$ e $I_{IL} = 0{,}4$ mA.

Solução

O carregamento na saída da porta 1 é equivalente a seis cargas de entrada 74LS no estado ALTO, mas a apenas cinco cargas de entrada no estado BAIXO. Isto é porque a porta NAND representa apenas uma única carga de entrada no estado BAIXO.

Colocando Entradas TTL em Nível BAIXO

Ocasionalmente, surgem situações nas quais uma entrada TTL deve ser mantida normalmente em BAIXO, e então deve ir para ALTO pela atuação de uma chave mecânica. Isto é ilustrado na Fig. 8-17 para a entrada de um monoestável. Este MONO é disparado por uma transição positiva que ocorre quando a chave é momentaneamente fechada. O resistor R serve para manter a entrada T em BAIXO enquanto a chave permanece aberta. Deve-se ter o cuidado de manter o valor de R baixo o suficiente para que a tensão sobre ele devido à corrente I_{IL} que flui da entrada do MONO para a terra não exceda V_{IL}(max). Assim, o maior valor de R é dado por

$$I_{IL} \times R_{max} = V_{IL}(\text{max})$$

$$R_{max} = \frac{V_{IL}(\text{max})}{I_{IL}} \qquad (8\text{-}3)$$

R deve ser mantido abaixo deste valor para garantir que a entrada do MONO estará com um nível BAIXO aceitável enquanto a chave estiver aberta. O valor mínimo de R é determinado pelo consumo de corrente da fonte de 5 V quando a chave for fechada. Na prática, esta corrente deve ser minimizada mantendo-se R ligeiramente abaixo de R_{max}.

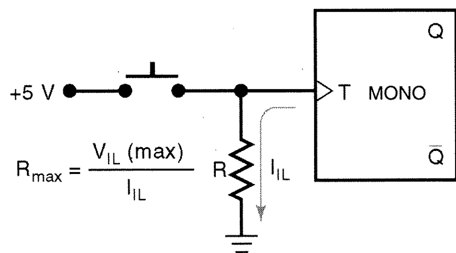


Fig. 8-17

EXEMPLO 8-11

Determine um valor aceitável para R se o MONO for um CI TTL 74LS com uma corrente I_{IL} de 0,4 mA.

Solução

O valor de I_{IL} será no máximo 0,4 mA. Este valor máximo seria usado para calcular R_{max}. Da Tabela 8-6, V_{IL}(max) = 0,8 V para a série 74LS. Assim, temos

$$R_{max} = \frac{0{,}8\ \text{V}}{0{,}4\ \text{mA}} = 2000\ \Omega$$

Uma boa escolha aqui seria $R = 1{,}8$ kΩ, um valor padrão de resistor.

Transientes de Corrente

Os circuitos lógicos TTL são afetados pelos spikes ou pelos transientes internos de corrente causados pela estrutura de saída totem-pole. Quando a saída está comutando do estado BAIXO para o estado ALTO (vide Fig. 8-18), os dois transistores de saída estão mudando de estado: Q_3 de OFF para ON, e Q_4 de ON para OFF. Tendo em vista que Q_4 está saindo da condição de saturação, ele leva mais tempo do que Q_3 para mudar de estado. Logo, existe um

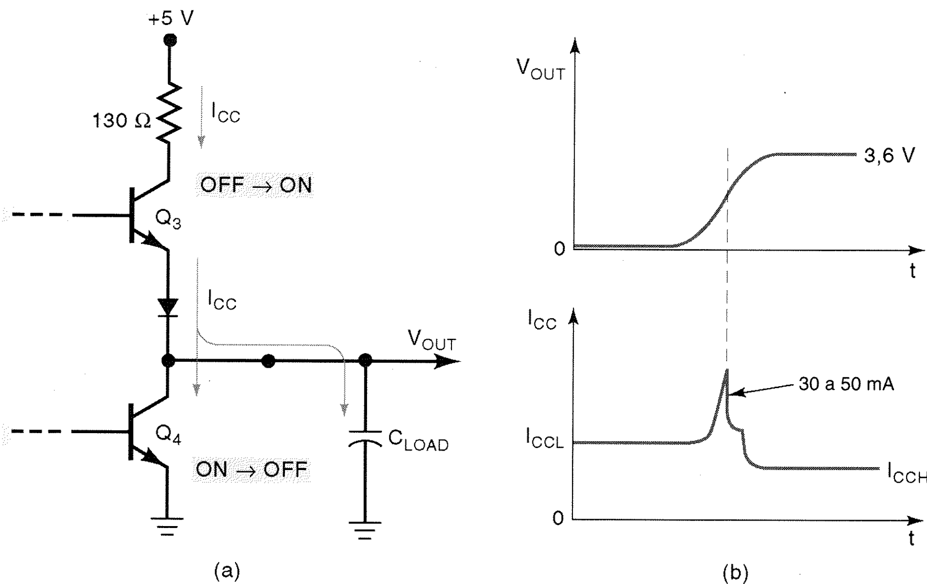


Fig. 8-18 - Um grande spike de corrente é consumido de V_{CC} quando uma saída totem-pole comuta de BAIXO para ALTO.

pequeno intervalo de tempo (por volta de 2 ns) durante a comutação em que ambos os transistores estão conduzindo e um surto (pico) de corrente relativamente alto (30 a 50 mA) é solicitado da fonte de +5 V. A duração deste transiente de corrente é estendida pelos efeitos de qualquer capacitância de carga no circuito de saída. Esta capacitância consiste em capacitâncias parasitas das ligações e em capacitâncias de entrada de quaisquer circuitos de carga, e deve ser carregada para o nível de tensão do estado de saída ALTO. Este efeito total pode ser resumido como se segue:

Sempre que uma saída TTL totem-pole vai de BAIXO para ALTO, um pico de corrente de alta amplitude é drenado da fonte V_{CC}.

Num circuito ou sistema digital complexo, existem muitas saídas TTL trocando de estado ao mesmo tempo, cada uma drenando um spike de corrente da fonte. O efeito cumulativo de todos estes picos de corrente será produzir um spike de tensão na linha V_{CC}, devido principalmente à indutância distribuída na linha da fonte de alimentação [lembre-se: $V = L(di/dt)$ para indutância, e di/dt é muito grande para um spike de corrente de 2 ns]. Este spike de tensão pode causar sérios problemas durante as transições, a menos que algum tipo de filtragem seja usado. A técnica mais comum utiliza pequenos capacitores de radiofreqüência conectados entre V_{CC} e TERRA, para essencialmente "curtar" estes spikes de alta freqüência. Isto é chamado **desacoplamento da fonte de alimentação**.

É prática comum conectar um capacitor de disco cerâmico de 0,01 μF ou 0,1 μF de baixa indutância entre V_{CC} e terra próximo de cada CI TTL em uma placa de circuito impresso. Os terminais do capacitor são mantidos bem pequenos para minimizar a indutância série.

Além disso, é procedimento comum conectar-se um grande capacitor (2 a 20 μF) entre V_{CC} e terra de cada placa para filtrar as variações de relativamente baixa freqüência em V_{CC} causadas pelas grandes mudanças nos níveis de I_{CC} à medida que as saídas comutam de estado.

Questões de Revisão

1. Qual será a saída lógica de uma porta NAND TTL que tem todas as suas entradas desconectadas?
2. Quais são duas maneiras aceitáveis de lidar com entradas não-utilizadas em portas AND?
3. Repita a questão 2 para uma porta NOR.
4. *Verdadeiro ou falso:* Quando as entradas de uma porta NAND são conectadas juntas, elas sempre são tratadas como uma única carga para a fonte de sinal.
5. O que é desacoplamento da fonte de alimentação? Por que ele é usado?

8-7 CONECTANDO SAÍDAS TTL JUNTAS

Por mais estranho que possa parecer, existem situações nas quais é vantajoso conectar as *saídas* de duas ou mais portas lógicas (ou outros dispositivos). Sempre que isto é feito, devemos estar cientes da situação em que uma saída está tentando ir para BAIXO, enquanto outra saída está tentando ir para ALTO; como estão ligadas juntas, temos um conflito ALTO/BAIXO. Como veremos em breve, com dispositivos TTL o BAIXO sempre ganha. Até agora, nos familiarizamos somente com a estrutura de saída totem-pole para dispositivos TTL. Dois outros tipos de estruturas de saída serão apresentados. Primeiramente vamos considerar a tentativa de ligar saídas totem-pole juntas.

Saídas totem-pole não devem ser ligadas juntas.

A razão para esta regra geral pode ser vista se observarmos a Fig. 8-19, onde as saídas totem-pole de duas portas são conectadas juntas no ponto *X*. Suponha que a saída da porta *A* está no estado ALTO (Q_{3A} ON, Q_{4A} OFF) e a saída da porta *B* está no estado BAIXO (Q_{3B} OFF, Q_{4B} ON). Nesta situação, Q_{4B} é uma resistência de carga muito baixa para Q_{3A} e consumirá uma corrente que pode ir até 55 mA. Esta cor-

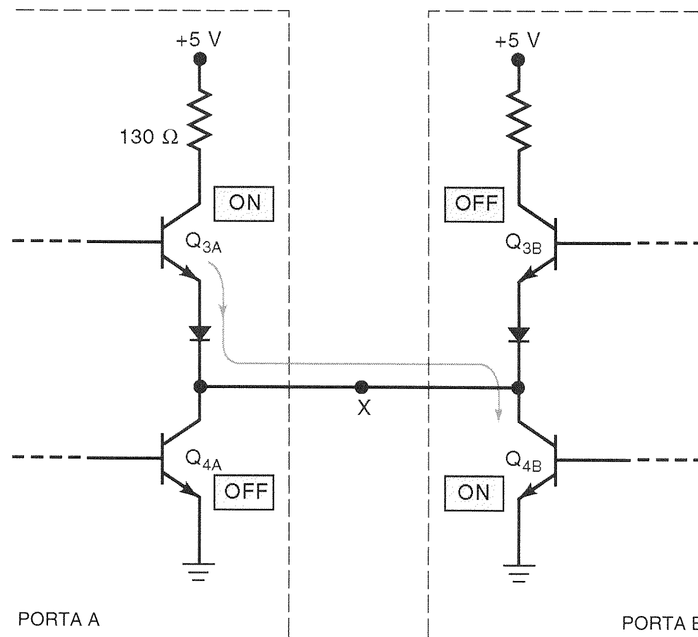


Fig. 8-19 - Saídas totem-pole ligadas juntas podem produzir uma corrente muita alta através de Q_4.

rente pode não danificar Q_{3B} nem Q_{4A} imediatamente, mas com o passar do tempo pode causar superaquecimento, deterioração de performance e falha eventual do dispositivo. A situação piora quando mais de duas saídas totem-pole são ligadas juntas.

Um outro problema causado por esta corrente relativamente alta fluindo através de Q_{4A} é que ela produzirá uma queda de tensão maior entre o coletor e o emissor do transistor, fazendo V_{OL} maior do que o V_{OL}(max) permitido.

Algumas vezes, saídas totem-pole são casualmente colocadas juntas por causa de erros de ligação ou curtos acidentais na placa de circuito impresso. Quando isto ocorre, o sinal no ponto comum usualmente será o AND lógico das saídas; isto é, ele estará BAIXO quando qualquer um dos sinais de saída colocados juntos for para BAIXO. Isto é verdade somente para circuitos TTL; MOS e CMOS se comportam de maneira menos previsível quando as saídas são conectadas juntas.

Saídas Coletor Aberto

Alguns circuitos TTL são projetados com **saídas coletor aberto**. Conforme mostrado na Fig. 8-20(a), a estrutura coletor aberto elimina o transistor de pull-up Q_3, D_1 e R_4. A saída é no coletor de Q_4, que está aberto (desconectado). No estado de saída BAIXO, Q_4 está ON (tem corrente de base e é essencialmente um curto entre coletor e emissor); no estado ALTO de saída, Q_4 está OFF (não tem corrente de base e está essencialmente aberto entre coletor e emissor). Para operação adequada, um resistor de pull-up externo, R_P, deve ser conectado como mostrado na Fig. 8-20(b). Este resistor não é parte do circuito interno do dispositivo TTL; ele é um resistor que você deve conectar na saída do dispositivo.

Quando Q_4 está ON, ele deixa a tensão de saída em BAIXO. Quando Q_4 está OFF, R_P faz a tensão de saída ir para ALTO. Note que sem o resistor de pull-up a tensão de saída seria indeterminada (flutuando). Por isso o resistor de pull-up é usado. O valor deste resistor é usualmente escolhido como 10 kΩ. Este valor é pequeno o bastante para que no estado ALTO a queda de tensão sobre ele, devido à corrente de carga, não abaixará a tensão de saída para menos do que o mínimo para TTL. Ele é grande o suficiente para que no estado BAIXO limite a corrente através de Q_4 para um valor abaixo de I_{OL}(max).

Conexão Wired-AND

Dispositivos com saídas em coletor aberto podem ter suas saídas conectadas juntas de modo seguro. A Fig. 8-21 mostra três portas NAND com coletor aberto (74LS01) cujas saídas são ligadas juntas. Esta conexão é denominada **wired-AND** porque é equivalente à operação lógica AND. A expressão lógica da saída é a mesma que seria obtida se as saídas das três portas tivessem sido ligadas em uma porta AND. Isto é mostrado pelo símbolo de porta AND pontilhado. Não existe nenhuma porta AND de verdade.

Esta configuração wired-AND elimina a necessidade de uma porta AND real, mas os dispositivos com coletor aberto apresentam uma velocidade de chaveamento bem menor do que aqueles com saída totem-pole, que têm um transistor de pull-up (Q_3) para carregar a capacitância de carga rapidamente. Por isso, os circuitos com coletor aberto não devem ser usados onde a velocidade é primordial.

Buffers/Drivers de Coletor Aberto

Todo circuito lógico que é chamado de *buffer*, *driver* ou **buffer/driver** é projetado para ter uma corrente e/ou uma

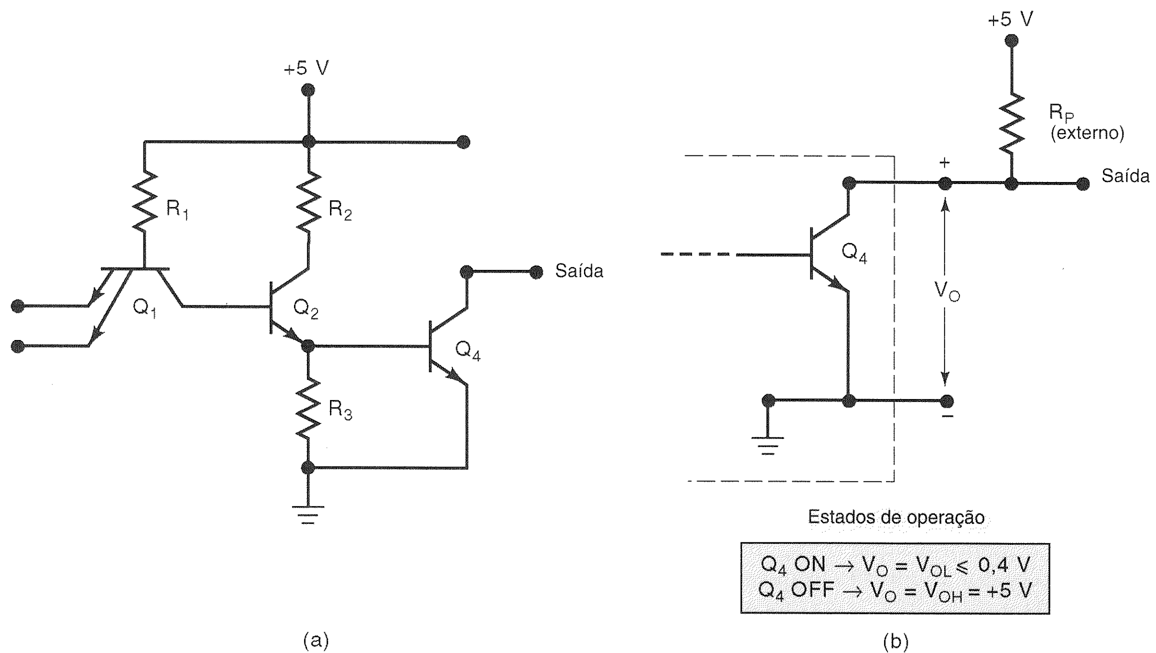


Fig. 8-20 - (a) Circuito TTL coletor aberto; (b) com resistor de pull-up externo.

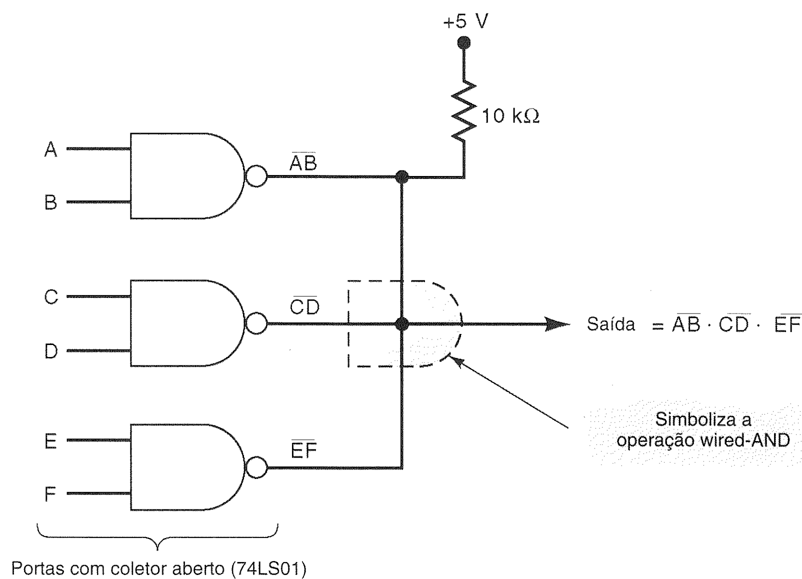


Fig. 8-21 - Operação wired-AND utilizando portas com coletor aberto.

capacidade de tensão na saída maior do que um circuito lógico comum. CIs buffers/drivers estão disponíveis com saídas totem-pole e com saídas em coletor aberto.

Devido a suas especificações de alta corrente e tensão de saída, o 7406 e o 7407 são os únicos dispositivos TTL padrão que ainda são recomendados para novos projetos. O 7406 é um CI buffer/driver de coletor aberto que contém seis INVERSORES com saídas em coletor aberto que podem absorver até 40 mA no estado BAIXO. Além disso, o 7406 aceita tensões de saída de até 30 V. Isto significa que o transistor de saída pode ser conectado a uma tensão maior do que 5 V.

Isto está ilustrado na Fig. 8-22, onde um 7406 é usado como um buffer entre um flip-flop 74LS112 e uma lâmpada incandescente de 24 V e 25 mA. O 7406 controla o estado ON/OFF da lâmpada para indicar o estado da saída Q do FF. Repare que a lâmpada é alimentada com +24 V e funciona como o resistor de pull-up para a saída de coletor aberto.

Quando $Q = 1$, a saída do 7406 vai para BAIXO, seu transistor de saída absorve os 25 mA de corrente da lâmpada fornecida pela fonte de 24 V e a lâmpada fica ligada. Quando $Q = 0$, o transistor de saída do 7406 desliga; não existe caminho para a corrente, e a lâmpada apaga. Neste estado,

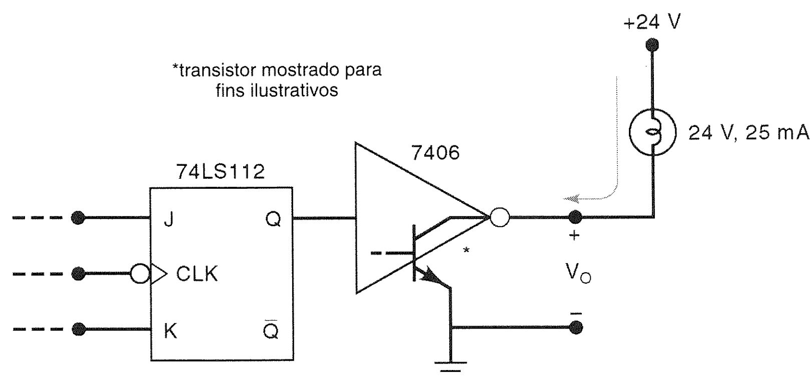


Fig. 8-22 - Um buffer/driver com coletor aberto aciona uma carga de alta corrente e alta tensão.

os 24 V aparecem sobre o transistor de saída, de modo que V_{OH} = 24 V, que é menor do que o V_{OH} máximo do 7406.

Saídas do tipo coletor aberto são freqüentemente usadas para acionar LEDs indicadores, como mostrado na Fig. 8-23. O resistor é usado para limitar a corrente em um valor seguro. Quando a saída do INVERSOR está em BAIXO, seu transistor de saída apresentará um caminho de baixa resistência para a corrente do LED para a terra, de modo que o LED ficará aceso. Quando a saída do INVERSOR está em ALTO, seu transistor de saída estará cortado, e não existirá caminho para a corrente do LED; neste estado o LED estará apagado.

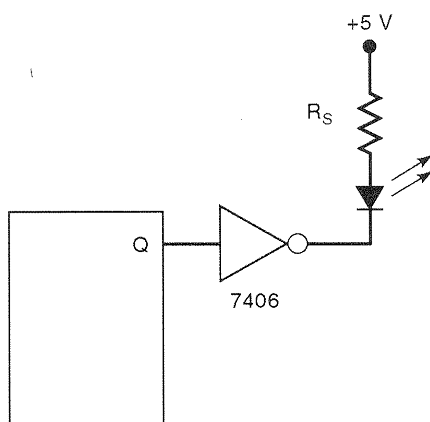


Fig. 8-23 - Uma saída coletor aberto pode ser usada para acionar um LED indicador.

O CI 7407 é buffer não-inversor com saídas em coletor aberto com as mesmas especificações de tensão e corrente do 7406.

Símbolo IEEE/ANSI para Saídas em Coletor Aberto

Os símbolos tradicionais para circuitos lógicos com saídas em coletor aberto são os mesmos que os de saídas totem-pole. A nova simbologia IEEE/ANSI, entretanto, utiliza uma notação diferente para identificar saídas de coletor aberto. A Fig. 8-24 mostra a designação padrão IEEE/ANSI para uma saída em coletor aberto. É um losango sublinhado. Apesar de não utilizarmos normalmente a simbologia IEEE/ANSI neste livro, utilizaremos este losango sublinhado para indicar saídas em coletor aberto.

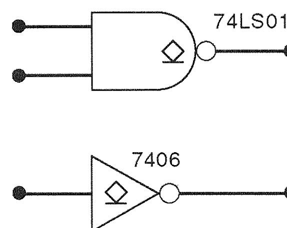


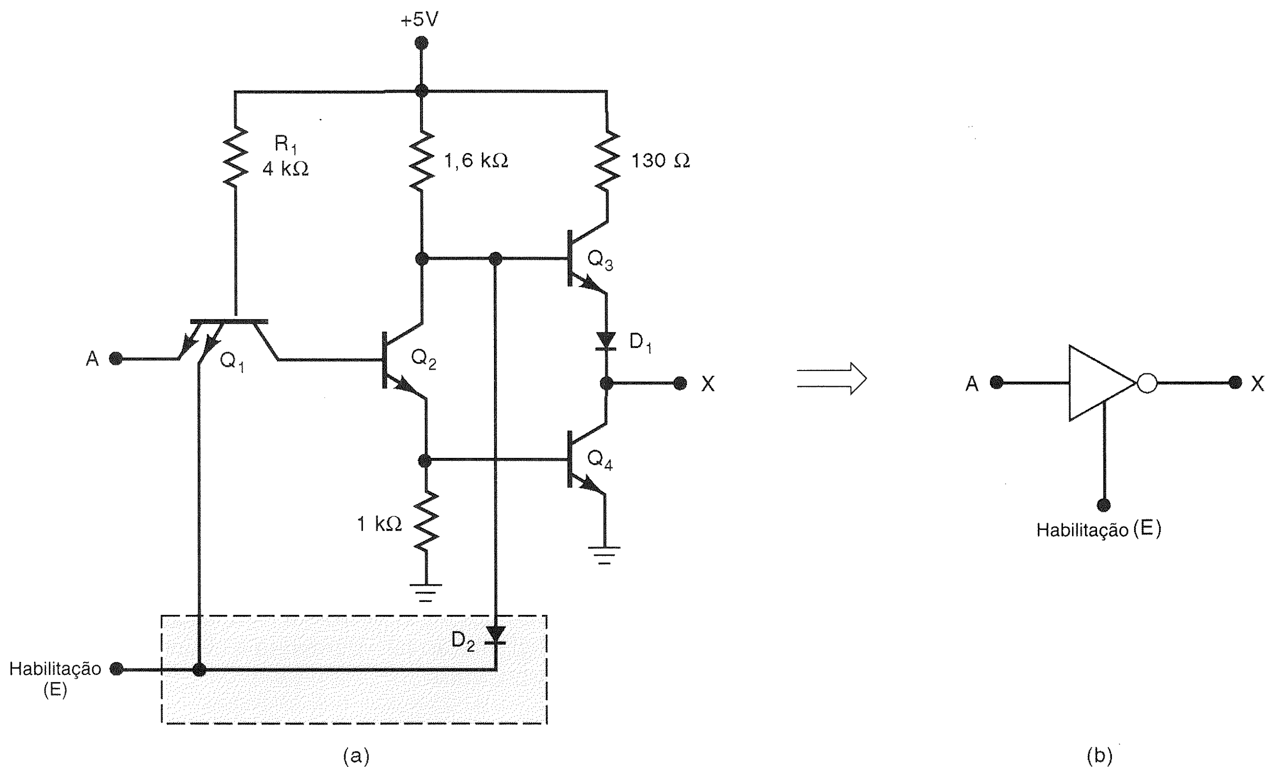
Fig. 8-24 - Notação IEEE/ANSI para saídas em coletor aberto.

Questões de Revisão

1. Quando ocorre um conflito ALTO/BAIXO?
2. Por que saídas totem-pole não devem ser ligadas juntas?
3. Como as saídas em coletor aberto diferem das saídas totem-pole?
4. Por que as saídas de coletor aberto precisam de um resistor de pull-up?
5. Qual é a expressão lógica para a conexão wired-AND de seis saídas 7406?
6. Por que as saídas em coletor aberto são geralmente mais lentas do que as totem-pole?
7. Qual é o símbolo IEEE/ANSI para saídas em coletor aberto?

8-8 TRISTATE (TERCEIRO ESTADO) PARA TTL

A configuração **tristate** é o terceiro tipo de configuração de saída TTL. Ela possui a operação de alta velocidade do arranjo totem-pole, enquanto permite que as saídas sejam conectadas juntas. É chamada de TTL *tristate* porque per-



Condições de operação

HABILITAÇÃO (E)	SAÍDA
ALTO	Habilitado: opera como um INVERSOR
BAIXO	Desabilitado: a saída está em alta impedância. A entrada A não tem efeito.

(c)

Fig. 8-25 - INVERSOR TTL tristate.

mite *três* estados de saída possíveis: ALTO, BAIXO e alta impedância (Hi-*Z*). O estado de alta impedância é uma condição na qual ambos os transistores do arranjo totem-pole estão cortados, e portanto o terminal de saída apresenta uma alta impedância para a terra e para V_{CC}. Em outras palavras, a saída é um terminal aberto, ou flutuante, que não está nem em BAIXO nem em ALTO. Na prática, o terminal de saída não é exatamente um circuito aberto, mas tem uma resistência de vários megaohms em relação a terra e a V_{CC}.

A operação tristate é obtida modificando-se o circuito totem-pole básico. A Fig. 8-25(a) mostra o circuito para um INVERSOR tristate onde o que está dentro do pontilhado foi incluído no circuito básico. O circuito tem duas entradas: *A* é a entrada lógica normal, e *E* é uma entrada de HABILITAÇÃO (ENABLE) que pode produzir o estado de alta impedância. Examinaremos a operação para ambos os estados de *E*.

Estado Habilitado

Com $E = 1$, o circuito opera como um INVERSOR normal, pois um nível ALTO em *E* não tem efeito sobre Q_1 nem D_2. Nesta condição habilitada, a saída simplesmente é o inverso da entrada lógica *A*.

Estado Desabilitado (Alta Impedância)

Quando $E = 0$, o circuito vai para o estado de alta impedância independentemente do estado da entrada lógica *A*. Um nível BAIXO em *E* polariza diretamente a junção base-emissor de Q_1 e desvia a corrente de R_1 da base de Q_2, de modo que Q_2 corta, o que deixa Q_4 cortado. O nível BAIXO em *E* polariza diretamente o diodo D_2 para desviar corrente da base de Q_3, de modo que Q_3 também corta.

Com ambos os transistores do arranjo totem-pole sem conduzir, o terminal de saída está praticamente em circuito aberto. Isto é mostrado simbolicamente na tabela da Fig. 8-25(c).

O símbolo lógico para o INVERSOR tristate é mostrado na Fig. 8-25(b). Repare onde a entrada de HABILITAÇÃO é colocada no símbolo do INVERSOR. Note também que E é ativo-ALTO; isto é, o INVERSOR está habilitado quando $E = 1$.

Vantagem do Tristate

As saídas de CIs tristate podem ser conectadas juntas (em paralelo) sem sacrificar a velocidade de chaveamento. Isto é porque a saída tristate, quando habilitada, opera como uma saída totem-pole, com suas características de baixa impedância e alta velocidade. É importante perceber, entretanto, que, quando saídas tristate estão em paralelo, apenas uma delas pode estar habilitada num certo instante. Senão, duas saídas totem-pole ativas estariam conectadas e correntes de alto valor poderiam fluir (Fig. 8-19). Vamos estudar melhor isto em nossa apresentação sobre buffers tristate.

Buffers Tristate

Um *buffer tristate* é um circuito usado para controlar a passagem de um sinal lógico de uma entrada para uma saída. Alguns buffers tristate também invertem o sinal. O circuito na Fig. 8-25 pode ser chamado de um *buffer tristate inversor.*

Dois CIs de buffers tristate usualmente utilizados são o 74LS125 e o 74LS126. Ambos contêm quatro buffers tristate *não-inversores,* como aqueles mostrados na Fig. 8-26. O 74LS125 e o 74LS126 diferem apenas no estado ativo de suas entradas de HABILITAÇÃO. O 74LS125 permite que o sinal A alcance a saída quando $\overline{E} = 0$, enquanto o 74LS126 permite quando $E = 1$.

Buffers tristate têm muitas aplicações em circuitos onde diversos sinais são conectados em linhas comuns (barramentos). Examinaremos algumas aplicações no Cap. 9, mas podemos captar a idéia básica a partir da Fig. 8-27(a). Temos três sinais lógicos, A, B e C, conectados em uma linha comum de barramento através de buffers tristate 74LS126. Este arranjo nos permite transmitir qualquer um destes sinais pela linha do barramento para outros circuitos, habilitando o buffer apropriado.

Por exemplo, considere a situação da Fig. 8-27(b), onde $E_B = 1$ e $E_A = E_C = 0$. Isto desabilita os buffers superior e inferior, de modo que suas saídas estão no estado de alta impedância e portanto estão essencialmente desconectadas do barramento. Isto é simbolizado pelos X's no diagrama. O buffer do meio está habilitado, e portanto sua entrada, B, vai para a sua saída e para o barramento, de onde é levada para outros circuitos conectados ao barramento. Quando saídas tristate são conectadas juntas, como na Fig. 8-27, é importante lembrar que apenas uma saída pode estar habilitada de cada vez. Caso contrário, duas ou mais saídas totem-pole ativas poderiam ser conectadas, o que poderia produzir correntes de alto valor. Mesmo que não ocorressem danos, esta situação produziria um sinal no barramento que seria a combinação de mais de um sinal. Isto é comumente

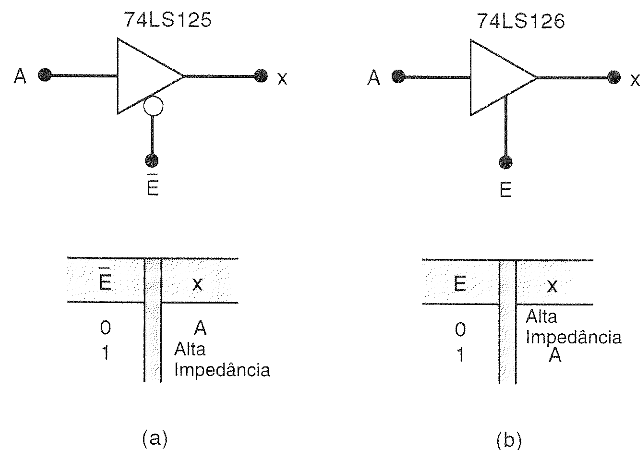


Fig. 8-26 - Buffers tristate não-inversores.

chamado **contenção de barramento.** Em sistemas com barramentos tristate, o projetista deve estar certo de que os sinais de habilitação não permitam a ocorrência de contenção de barramento.

CIs Tristate

Além dos buffers tristate, muitos CIs são projetados com saídas de terceiro estado. Por exemplo, o 74LS374 é um CI registrador com oito FFs do tipo D com saídas tristate. Isto significa que ele é um registrador de oito bits construído com FFs do tipo D cujas saídas estão conectadas a buffers de terceiro estado. Este tipo de registrador pode ser conectado a linhas de barramento juntamente com as saídas de outros dispositivos similares, para permitir uma transferência de dados eficiente através do barramento. Examinamos este arranjo de *barramento de dados tristate* no Cap. 9. Outros tipos de dispositivos lógicos que estão disponíveis com saídas em terceiro estado incluem os decodificadores, multiplexadores, conversores analógico-digitais, chips de memória e microprocessadores.

Símbolo IEEE/ANSI para Saídas Tristate

A simbologia lógica tradicional não tem uma notação especial para saídas de terceiro estado. A Fig. 8-28 mostra a notação usada na simbologia IEEE/ANSI para indicar uma saída tristate. É um triângulo que aponta para baixo. Embora não seja parte da simbologia tradicional, no restante deste livro utilizaremos este triângulo para indicar as saídas tristate.

Questões de Revisão

1. Quais são os três estados possíveis de saída de um CI tristate?
2. Qual é o estado de uma saída tristate quando está desabilitada?
3. O que é contenção de barramento?
4. Que condições são necessárias para transmitir o sinal C pelo barramento, na Fig. 8-27?
5. Qual é a notação IEEE/ANSI para saídas tristate?

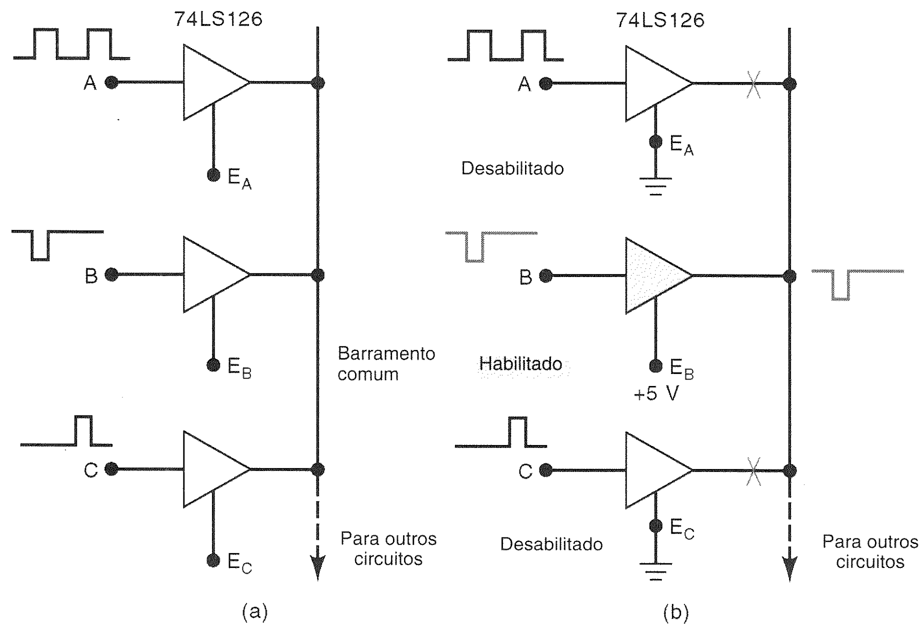


Fig. 8-27 - (a) Buffers tristate usados para conectar diversos sinais em um barramento comum; (b) condições para a transmissão de *B* para o barramento.

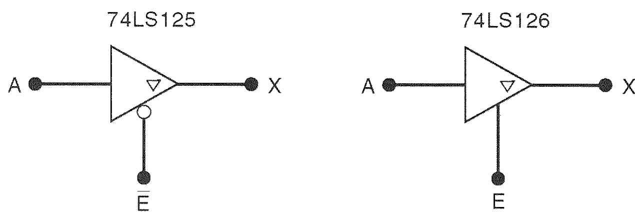


Fig. 8-28 - Notação IEEE/ANSI para saídas tristate.

8-9 A FAMÍLIA ECL DE CIs DIGITAIS

A família TTL utiliza transistores operando na saturação. Em resultado, sua velocidade de comutação está limitada pelo atraso do tempo de armazenamento associado com o transistor que está saturado. Uma outra família lógica *bipolar* foi desenvolvida evitando a saturação, e portanto aumentando a velocidade global de chaveamento. Esta família lógica é chamada **lógica com acoplamento pelo emissor** (**ECL** — *emitter-coupled logic*), e ela opera sob o princípio do chaveamento de corrente, onde uma corrente fixa de polarização menor do que I_C(sat) é chaveada do coletor de um transistor para outro. Devido a este modo de operação por corrente, esta forma de lógica também é conhecida como *lógica em modo de corrente.*

Circuito Básico ECL

O circuito básico para a família ECL é essencialmente a configuração de amplificador diferencial da Fig. 8-29(a). A fonte de alimentação V_{EE} produz uma corrente I_E praticamente fixa por volta de 3 mA durante a operação normal. Esta corrente pode fluir através de Q_1 ou Q_2, dependendo do nível de tensão em V_{IN}. Em outras palavras, esta corrente será chaveada entre o coletor de Q_1 e o coletor de Q_2 conforme V_{IN} comuta entre seus dois níveis lógicos de −1,7 V (0 lógico para ECL) e −0,8 V (1 lógico para ECL). A tabela na Fig. 8-29(a) mostra as tensões de saída resultantes para estas duas condições em V_{IN}. Dois pontos importantes devem ser notados: (1) V_{C1} e V_{C2} são o *complemento* um do outro, e (2) os níveis de tensão de saída não são os mesmos dos níveis lógicos de entrada.

O segundo ponto indicado é facilmente resolvido conectando V_{C1} e V_{C2} em estágios do tipo seguidor de emissor (Q_3 e Q_4), conforme mostrado na Fig. 8-29(b). Os seguidores de emissor realizam duas funções: (1) subtraem aproximadamente 0,8 V de V_{C1} e V_{C2} para deslocar os níveis de saída para os níveis lógicos corretos ECL; e (2) têm uma impedância de saída muito baixa (geralmente 7 Ω), que proporciona um grande fan-out e um rápido carregamento para capacitâncias de carga. Este circuito produz duas saídas complementares: V_{OUT1}, que é igual a $\overline{V_{IN}}$, e V_{OUT2}, que é igual a V_{IN}.

Porta OR/NOR ECL

O circuito ECL básico da Fig. 8-29(b) pode ser usado como um INVERSOR se a saída for tomada em V_{OUT1}. Este circuito básico pode ser expandido para mais de uma entrada, colocando outros transistores em paralelo com Q_1 para outras entradas, conforme a Fig. 8-30(a). Aqui, tanto Q_1 como Q_3 podem provocar o chaveamento da corrente desviando-a de Q_2, resultando nas duas saídas V_{OUT1} e V_{OUT2}, que são as operações NOR e OR, respectivamente. Esta porta OR/NOR está simbolizada na Fig. 8-30(b) e é a porta ECL fundamental.

Características ECL

A seguir estão as características mais importantes da família de circuitos lógicos ECL:

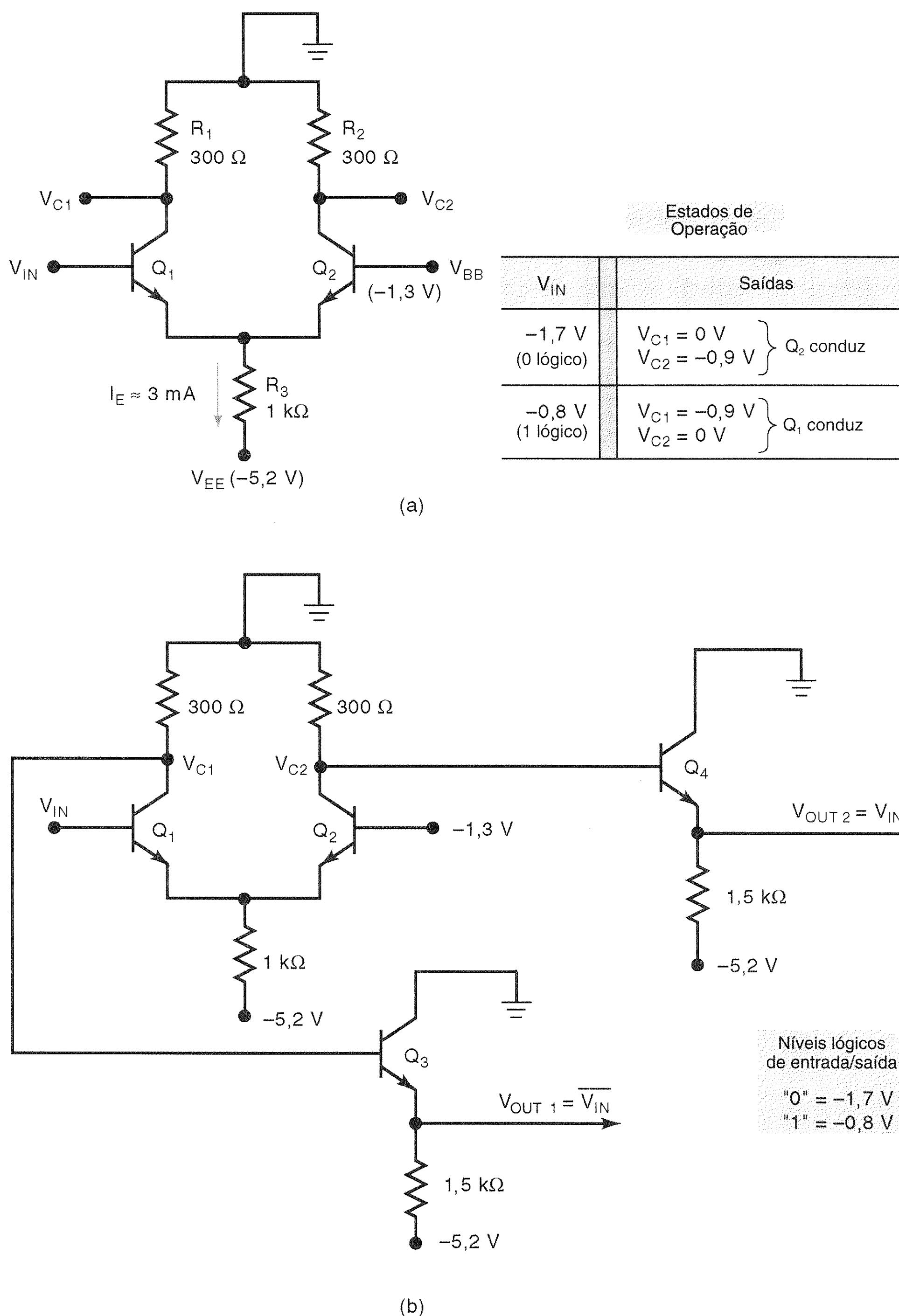


Fig. 8-29 - (a) Circuito ECL básico; (b) com a adição de seguidores de emissor.

1. Os transistores nunca saturam, e portanto a velocidade de comutação é muito alta. O atraso de propagação típico é de 1 ns, o que faz a família ECL ser um pouco mais rápida do que a TTL Schottky avançada (série 74AS).
2. Os níveis lógicos nominais são −0,8 V e −1,70 V para 1 e 0 lógicos, respectivamente.
3. As margens de ruído ECL para pior caso são de aproximadamente 250 mV. Estas pequenas margens de ruído tornam ECL não muito confiável para o uso em ambientes industriais.
4. Usualmente um bloco lógico ECL produz uma saída e seu complemento. Isto elimina a necessidade de inversores.
5. Fan-outs típicos são em torno de 25, graças às saídas de baixa impedância dos seguidores de emissor.
6. A dissipação típica de potência para uma porta básica ECL é 25 mW, um pouco mais alta do que a da série 74AS.
7. A corrente total em um circuito ECL permanece relativamente constante, não importando seu estado lógico. Isto ajuda a manter um consumo de corrente invariante na fonte de alimentação do circuito, mesmo durante as tran-

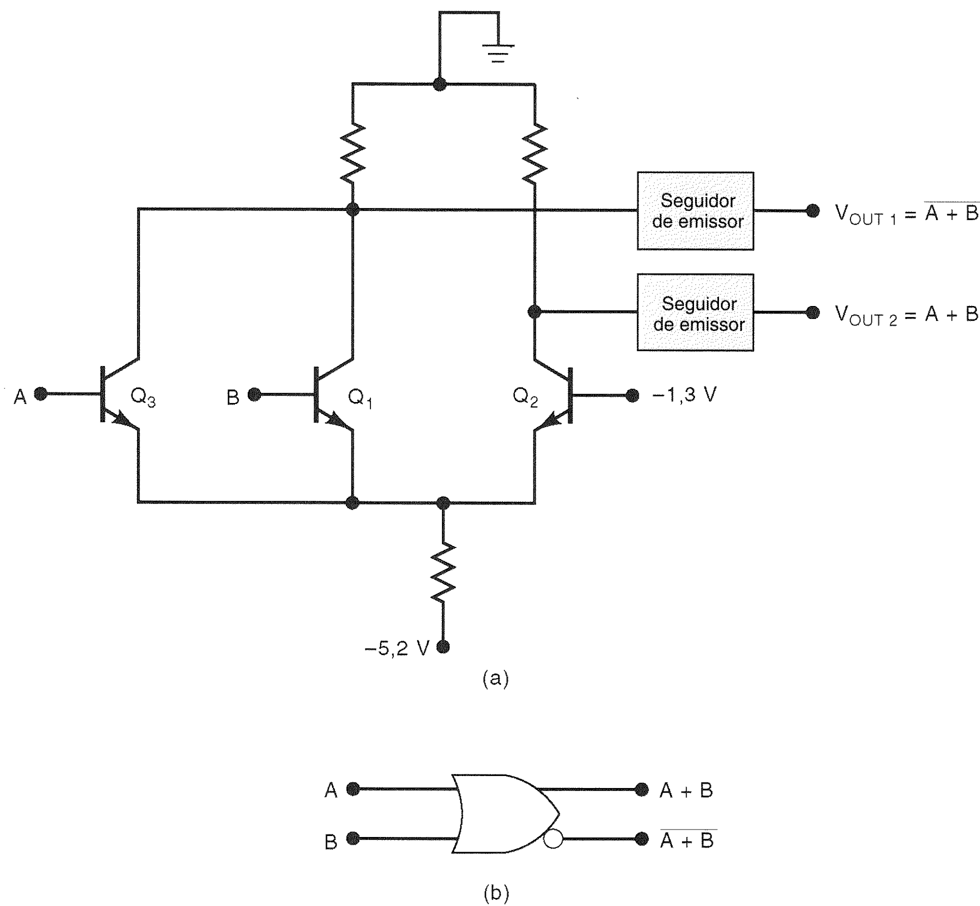


Fig. 8-30 - (a) Circuito NOR/OR ECL; (b) símbolo lógico.

sições de comutação. Assim, nenhum spike de ruído é gerado internamente, como aqueles produzidos pelos circuitos TTL totem-pole.

A Tabela 8-8 mostra como ECL se compara com as famílias lógicas TTL importantes. A família ECL de CIs não inclui uma ampla faixa de dispositivos lógicos de propósito geral, como as famílias TTL e CMOS. ECL inclui CIs complexos e de propósito especial, usados em aplicações de alta velocidade como transmissão de dados, memórias e unidades aritméticas. As baixas margens de ruído e o alto consumo de potência do ECL são desvantagens em comparação com TTL e CMOS. Outras desvantagens são sua fonte de alimentação e níveis lógicos negativos, que não são compatíveis com os de outras famílias lógicas. Isto torna difícil utilizar dispositivos ECL em conjunto com CIs TTL e/ou CMOS; circuitos especiais de deslocamento de níveis devem ser conectados entre dispositivos ECL e TTL (ou CMOS) tanto na entrada quanto na saída.

Questões de Revisão

1. *Verdadeiro ou falso*:
- **(a)** ECL consegue operar em alta velocidade evitando a saturação de transistores.
- **(b)** Os circuitos ECL usualmente têm saídas complementares.
- **(c)** As margens de ruído para os circuitos ECL são maiores do que as margens de ruído TTL.

TABELA 8-8

Família Lógica	t_{pd} (ns)	P_D (mW)	Margem de Ruído no Pior Caso (mV)	Taxa Máxima de Clock (MHz)
74	9	10	400	35
74AS	1,7	8	300	200
74ALS	4	1,2	400	70
74S	3	20	300	125
74LS	9,5	2	300	45
74F	3	6	300	100
ECL	1	25	250	600

(d) Circuitos ECL não geram spikes de ruído durante as transições de estado.
(e) Dispositivos ECL consomem menos potência do que os TTL padrão.
(f) ECL pode ser facilmente usado com TTL.

8-10 CIRCUITOS INTEGRADOS DIGITAIS MOS

A tecnologia MOS (*metal oxide semiconductor*— semicondutor com óxido metálico) tem o seu nome derivado da estrutura básica MOS, um eletrodo de metal sobre um óxido isolador, que por sua vez está sobre um substrato semicondutor. Os transistores da tecnologia MOS são transistores de efeito de campo chamados **MOSFETs**. Isto significa que o *campo* elétrico do lado do eletrodo de *metal* do *óxido* isolador tem *efeito* na resistência do substrato. A maioria dos CIs digitais MOS é constituída inteiramente de MOSFETs e de nenhum outro componente.

As vantagens principais do MOSFET são que ele é relativamente simples e barato para fabricar, é pequeno e consome pouca potência. A fabricação de CIs MOS tem aproximadamente um terço da complexidade da fabricação de CIs bipolares (TTL, ECL etc). Além disso, os dispositivos MOS ocupam bem menos espaço em um chip do que transistores bipolares; tipicamente, um MOSFET requer 1 milésimo de polegada quadrada de área de chip, enquanto um transistor bipolar requer por volta de 50. Mais importante, CIs digitais MOS normalmente não usam resistores integrados que ocupam bastante área de chip em CIs bipolares.

Tudo isto significa que os CIs MOS podem acomodar um número bem maior de elementos de circuitos em um único chip do que os CIs bipolares. Esta vantagem é evidente pelo fato de os CIs MOS terem dominado os CIs bipolares na área de integração em larga escala (LSI, VLSI). A alta densidade de encapsulamento dos CIs MOS os torna especialmente adequados para CIs complexos, tais como chips de microprocessadores e memórias. Avanços na tecnologia de CIs MOS têm levado a dispositivos que são mais rápidos do que os TTL 74, 74LS e 74ALS, com características de acionamento de corrente comparáveis. Conseqüentemente, os dispositivos MOS (especificamente os CMOS) também têm se tornado dominantes no mercado SSI e MSI. A família TTL 74AS ainda é mais rápida do que os melhores dispositivos MOS, mas com uma dissipação de potência bem maior.

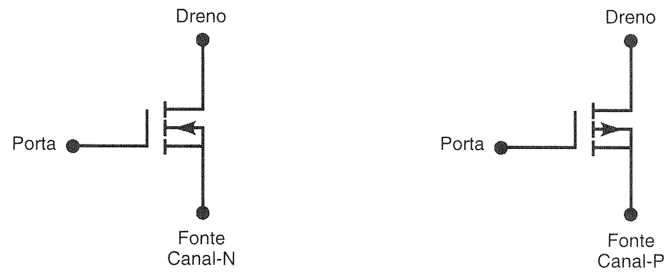


Fig. 8-31 - Símbolos de MOSFETs por enriquecimento.

A principal desvantagem dos dispositivos MOS é sua susceptibilidade a danos provocados por eletricidade estática. Embora isto possa ser minimizado com procedimentos de manuseio adequados, TTL ainda é mais resistente para experimentações em laboratório. Assim, enquanto estiverem disponíveis, é bem provável que você veja dispositivos TTL sendo usados no aprendizado de eletrônica digital.

8-11 O MOSFET

Existem atualmente dois tipos de MOSFET: por *deplexão* e por *enriquecimento*. Os CIs digitais MOS utilizam apenas MOSFETs por enriquecimento, e, portanto, somente este tipo será considerado na discussão a seguir. Além disso, iremos nos restringir a analisar o funcionamento destes MOSFETs como chaves liga/desliga.

A Fig. 8-31 mostra os símbolos esquemáticos para MOSFETs por enriquecimento canal-N e canal-P. A direção da seta indica o tipo de canal. Os símbolos mostram uma linha tracejada entre a *fonte* e o *dreno* para indicar que, *normalmente*, não há condução entre estes dois eletrodos. O símbolo também mostra a separação entre a *porta* e os outros terminais, para indicar a resistência muito alta (geralmente em torno de 10^{12} Ω) da camada de óxido entre a porta e o canal, que é formada no substrato.

Configuração Básica de um MOSFET como Chave

A Fig. 8-32 mostra a operação de comutação de um *MOSFET* canal-N. Para um dispositivo com canal-N, o dreno está sempre polarizado positivamente em relação à fonte. A ten-

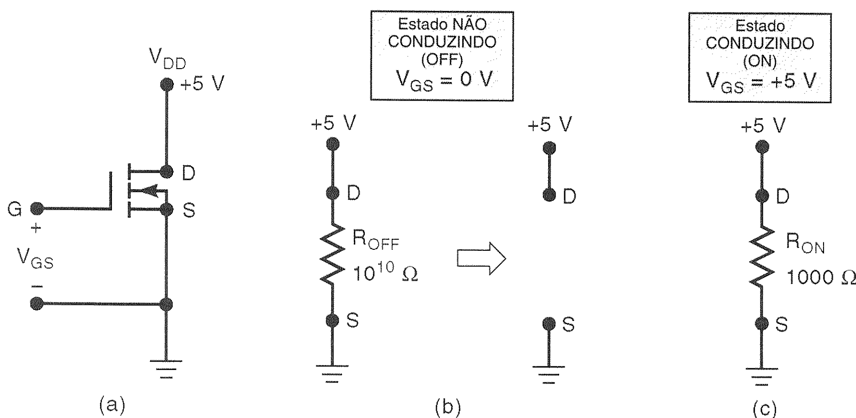


Fig. 8-32 - Estados de comutação do MOSFET canal-N.

são entre a porta e fonte, V_{GS}, é a tensão de entrada que é usada para controlar a resistência entre dreno e fonte (isto é, a resistência do canal), e portanto determina se o dispositivo está conduzindo ou não.

Quando V_{GS} = 0 V, não existe canal condutor entre fonte e dreno, e portanto o dispositivo não está conduzindo. Tipicamente a resistência do canal no estado OFF é de 10^{10} Ω, o que para a maioria dos casos representa um *circuito aberto*. O MOSFET permanecerá desligado enquanto V_{GS} for igual a zero ou for negativa. À medida que V_{GS} se torna positiva (tensão da porta positiva em relação à fonte), a tensão de limiar (V_T) é alcançada, e neste ponto um canal condutor começa a se formar entre fonte e dreno. Tipicamente, V_T = +1,5 V para um MOSFET-N e, portanto, um $V_{GS} \geq$ 1,5 V fará com que o MOSFET conduza. De um modo geral, um valor de V_{GS} muito maior do que V_T é usado para fazer com que o MOSFET conduza melhor. Conforme pode ser visto na Fig. 8-32(b), quando V_{GS} = +5 V, a resistência do canal entre fonte e dreno cai para um valor de R_{ON} = 1.000 Ω.

Em essência, o MOSFET-N vai comutar de uma resistência alta para uma resistência baixa, à medida que a tensão da porta vai de BAIXO para ALTO. É bastante útil pensar no MOSFET como uma chave que está aberta ou fechada entre fonte e dreno.

O MOSFET canal-P funciona do mesmo modo que o canal-N, exceto pelo fato de que ele usa tensão de polaridade inversa. Para MOSFETs canal-P, o dreno está conectado a $-V_{DD}$ e, portanto, está polarizado negativamente em relação à fonte. Para que um MOSFET-P conduza, uma tensão negativa que exceda V_T deve ser aplicada na porta.

A Tabela 8-9 resume as características de comutação do canal-P e do canal-N.

8-12 CIRCUITOS DIGITAIS COM MOSFETs

Circuitos digitais que utilizam MOSFETs são divididos em três categorias: (1) **P-MOS**, que utiliza *apenas* MOSFETs por enriquecimento com canal-P; (2) **N-MOS**, que utiliza *apenas* MOSFETs por enriquecimento com canal-N; e (3) **CMOS** (MOS complementar), que usa tanto dispositivos de canal-P quanto de canal-N.

CIs digitais P-MOS e N-MOS têm uma densidade de integração maior (mais transistores por chip) e portanto são mais econômicos que os CMOS. Os dispositivos N-MOS têm uma densidade de integração duas vezes maior que os P-MOS. Além disso, os N-MOS são cerca de duas vezes mais rápidos que os P-MOS. Isto acontece devido ao fato de que os elétrons livres são os portadores de corrente nos N-MOS, enquanto os buracos (cargas positivas mais lentas) são os portadores de corrente para os P-MOS. Os dispositivos CMOS têm a maior complexidade e a menor densidade de integração das famílias MOS, mas possuem vantagens importantes, como a alta velocidade e a baixa dissipação de potência. CIs N-MOS e CMOS são amplamente utilizados na área digital, mas CIs P-MOS não se encontram disponíveis para novos projetos. Entretanto, os MOSFETs de canal-P ainda são importantes porque são usados em circuitos CMOS.

Nesta seção vamos estudar alguns dos circuitos lógicos básicos N-MOS. Fica compreendido que o circuito P-MOS correspondente seria o mesmo, exceto pelo fato de que as tensões teriam polaridades diferentes. Uma vez que dispositivos N-MOS são mais amplamente utilizados em circuitos LSI e VLSI (microprocessadores, memórias etc.), vamos postergar qualquer aplicação de circuitos N-MOS para mais adiante. Os dispositivos CMOS, que, do mesmo modo que os TTL, são usados em circuitos MSI, serão estudados em detalhe no início da Seção 8-14.

Inversor N-MOS

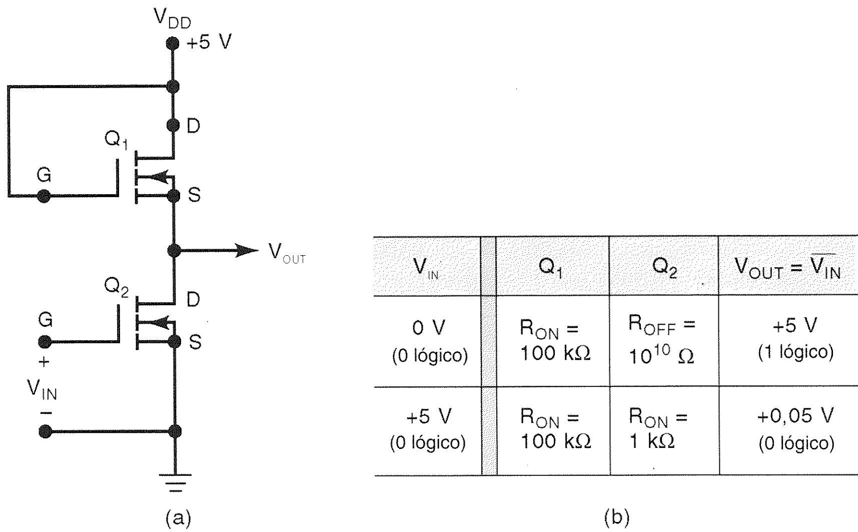
A Fig. 8-33 mostra o circuito básico de um INVERSOR N-MOS. Ele contém dois MOSFETs canal-N. Q_1 é chamado MOSFET *de carga*, e Q_2, MOSFET *de comutação*. Q_1 tem sua porta *permanentemente* conectada a +5 V, e portanto está *sempre* CONDUZINDO (ON) e funciona essencialmente com um resistor de carga com valor R_{ON}. Q_2 vai comutar de LIGADO (ON) para DESLIGADO (OFF) em resposta a V_{IN}. O MOSFET Q_1 é feito para ter um canal mais estreito do que Q_2, e portanto a R_{ON} de Q_1 é muito maior do que a de Q_2. Tipicamente, R_{ON} para Q_1 é 100 kΩ, e R_{ON} de Q_2 é 1 kΩ. R_{OFF} para Q_2 é geralmente em torno de 10^{10} Ω.

Os dois estados do INVERSOR estão resumidos na Fig. 8-33(b). A melhor maneira de analisar este circuito é considerar cada canal do MOSFET como uma resistência, de modo que a tensão de saída é obtida do divisor de tensão formado por estas duas resistências. Quando V_{IN} = 0 V, o transistor Q_2 está OFF, com uma resistência muito alta no canal de 10^{10} Ω. Uma vez que Q_1 tem um R_{ON} = 100 kΩ, a saída do divisor de tensão será essencialmente +5 V. Quando V_{IN} = +5 V, Q_2 está ON, com R_{ON} = 1 kΩ. O divisor de tensão é agora 100 kΩ e 1 kΩ, e portanto $V_{OUT} = 1/101 \times (+5\text{ V}) \approx$ 0,05 V.

O circuito é um INVERSOR porque uma entrada em nível BAIXO produz uma saída em nível ALTO, e vice-versa. Este INVERSOR básico pode ser modificado para formar portas lógicas NAND e NOR.

TABELA 8-9

	Polarização de Dreno para Fonte	Tensão entre Porta e Fonte (V_{GS}) Necessária para Condução	R_{ON} (Ω)	R_{OFF}(Ω)
Canal-P	Negativa	Geralmente mais negativa que − 1,5 V	1000 (típico)	10^{10}
Canal-N	Positiva	Geralmente mais positiva que + 1,5 V	1000 (típico)	10^{10}

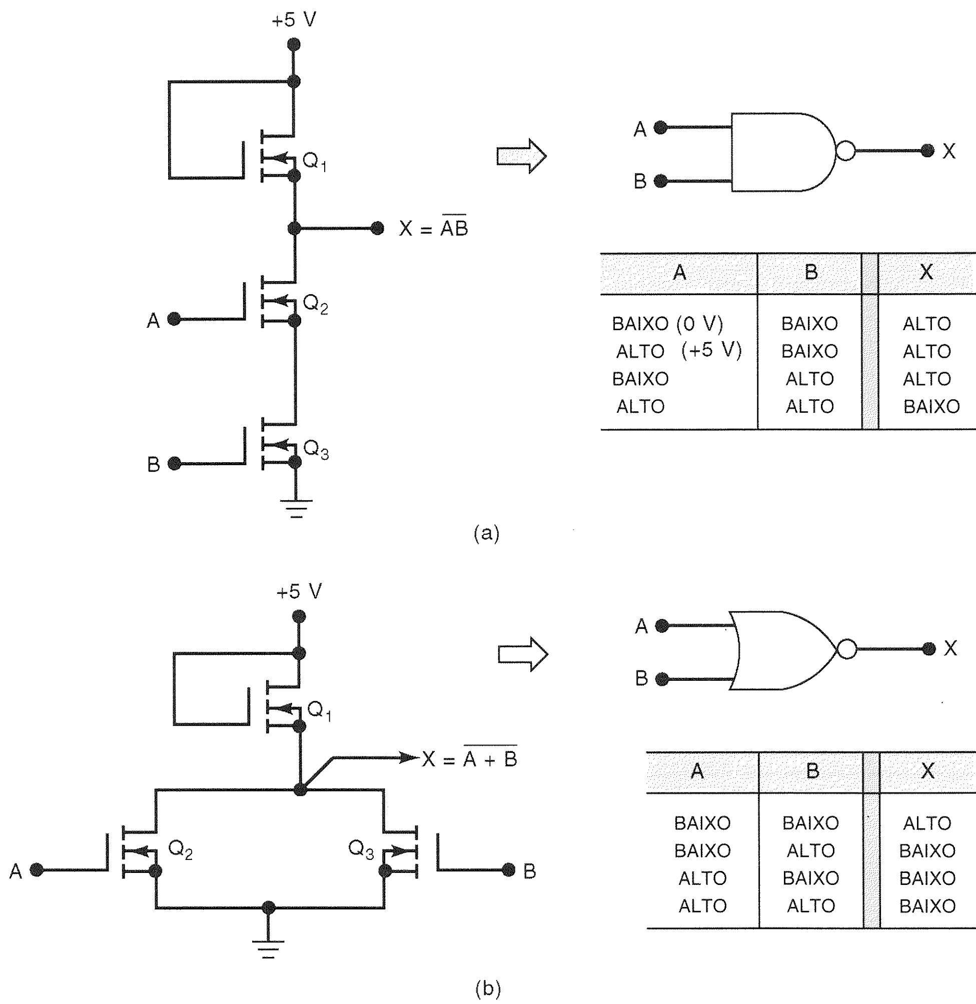


V_{IN}	Q_1	Q_2	$V_{OUT} = \overline{V_{IN}}$
0 V (0 lógico)	R_{ON} = 100 kΩ	R_{OFF} = 10^{10} Ω	+5 V (1 lógico)
+5 V (0 lógico)	R_{ON} = 100 kΩ	R_{ON} = 1 kΩ	+0,05 V (0 lógico)

Fig. 8-33 - INVERSOR N-MOS.

Porta NAND N-MOS

A operação NAND é realizada pelo circuito da Fig. 8-34(a), onde Q_1 está novamente funcionando como uma resistência de carga enquanto Q_2 e Q_3 são chaves controladas pelos níveis de entrada *A* e *B*. Se *A* ou *B* estiver com 0 V (0 lógico), o FET correspondente estará desligado, e apresentará uma resistência alta do terminal de saída para a terra, fazendo com que a saída *X* esteja em ALTO (+5 V). Quando ambas as entradas *A* e *B* estão com +5 V (1 lógico), Q_2 e Q_3



A	B	X
BAIXO (0 V)	BAIXO	ALTO
ALTO (+5 V)	BAIXO	ALTO
BAIXO	ALTO	ALTO
ALTO	ALTO	BAIXO

A	B	X
BAIXO	BAIXO	ALTO
BAIXO	ALTO	BAIXO
ALTO	BAIXO	BAIXO
ALTO	ALTO	BAIXO

Fig. 8-34 - (a) Porta NAND N-MOS; (b) porta NOR.

estarão ambos conduzindo, e portanto X estará em BAIXO. A saída é, obviamente, o resultado da operação NAND sobre as entradas ($X = \overline{AB}$).

Porta NOR N-MOS

A porta NOR da Fig. 8-34(b) usa Q_2 e Q_3 como chaves em paralelo, e Q_1 como resistência de carga. Quando ou a entrada A ou a B estiver com +5 V, o MOSFET correspondente estará ligado, e forçará a saída a ir para BAIXO. Quando ambas as entradas estão com 0 V, tanto Q_2 quanto Q_3 estão desligados, fazendo com que a saída vá para ALTO. Esta é, obviamente, a operação NOR sobre as entradas A e $B (X = \overline{A + B})$.

As portas OR N-MOS e as portas AND são facilmente formadas combinando-se NOR ou NAND com INVERSORes.

Flip-Flops N-MOS

Duas portas NOR N-MOS ou duas portas NAND podem ser acopladas de modo cruzado para formar latches simples (veja Figs. 5-3 e 5-10). Circuitos adicionais podem ser acrescentados para converter este latch básico em um flip-flop com clock J-K, ou um flip-flop com clock D (veja Fig. 5-23). Flip-flops N-MOS são importantes em alguns tipos de circuitos de memória.

8-13 CARACTERÍSTICAS DA LÓGICA MOS

Quando comparadas com famílias lógicas bipolares, as famílias lógicas N-MOS e P-MOS têm velocidade de operação menor; necessitam de muito menos potência; têm uma margem de ruído melhor, possuem uma faixa maior para a tensão de alimentação, um fan-out maior e, como foi mencionado anteriormente, necessitam de menos espaço (área de chip).

Velocidade de Operação

Uma típica porta NAND N-MOS tem um atraso de propagação de 50 ns. Isto é devido a *dois* fatores: (1) a resistência de saída relativamente alta (100 kΩ) no estado ALTO e (2) a carga capacitiva representada pelas entradas dos circuitos lógicos que estão sendo acionados. As entradas lógicas MOS têm uma resistência de entrada muito alta ($> 10^{12}$ Ω), e possuem uma capacitância de entrada relativamente alta (capacitor MOS), geralmente de 2 a 5 picofarads. Esta combinação de R_{OUT} alta com C_{LOAD} alta faz com que o tempo de comutação aumente.

Margem de Ruído

Tipicamente, as margens de ruído para a família N-MOS estão em torno de 1,5 V quando V_{DD} = 5 V, e serão proporcionalmente maiores para valores mais altos de V_{DD}.

Fan-Out

Devido à resistência de entrada extremamente alta em cada entrada MOSFET, poder-se-ia esperar que o fan-out da lógica CMOS fosse praticamente ilimitado. Isto é verdade apenas para a operação DC ou em baixas freqüências. Entretanto, para freqüências maiores que 100 kHz, as capacitâncias de entrada da porta causam uma deterioração no tempo de comutação, que aumenta na proporção do número de cargas que estão sendo acionadas. Mesmo assim, a lógica MOS pode facilmente operar com um fan-out de 50, que é muito melhor do que a maioria das famílias bipolares.

Consumo de Potência

Circuitos lógicos MOS consomem pequenas quantidades de potência por causa das resistências relativamente altas que são usadas. Para ilustrar este fato, vamos calcular a dissipação de potência do INVERSOR da Fig. 8-33 para os seus dois estados de operação:

1. **V_{IN} = 0 V:** $R_{ON(Q1)}$ = 100 kΩ; $R_{OFF(Q2)} = 10^{10}$ Ω. Portanto, I_D (corrente oriunda da fonte V_{DD}) ≈ 0,5 nA, e P_D = 5 V × 0,5 nA = 2,5 nW.
2. **V_{IN} = +5 V:** $R_{ON(Q1)}$ = 100 kΩ; $R_{ON(Q2)}$ = 1 kΩ. Portanto, I_D = 5 V/101 kΩ ≈ 50 μA, e P_D = 5 V × 50 μA = 0,25 mW.

Isto fornece uma P_D *média* um pouco maior que 0,1 mW para o INVERSOR. O baixo consumo da lógica MOS torna-a adequada a circuitos LSI e VLSI, onde muitas portas, FFs e outros circuitos podem estar presentes em um chip sem causar superaquecimento que poderia danificá-lo.

Complexidade do Processo de Fabricação

A família lógica MOS tem o processo de fabricação mais simples, porque utiliza apenas um elemento básico, o transistor N-MOS (ou P-MOS). Ela não precisa de nenhum outro elemento, como por exemplo resistores ou diodos. Esta característica, juntamente com sua baixa dissipação de potência, torna-a a mais indicada para circuitos LSI (grandes memórias, chips para calculadoras, microprocessadores), e foi nesta área que a lógica MOS teve maior impacto. A velocidade de operação da família P-MOS e da N-MOS não é comparável à da família TTL, portanto muito pouco foi feito com as famílias MOS em aplicações SSI e MSI. Na verdade, existem muito poucos circuitos MOS nas categorias MSI ou SSI (portas, FFs e contadores, por exemplo). A família CMOS, entretanto, é competitiva na área de circuitos MSI, que até bem pouco tempo era dominada pela família TTL.

Sensibilidade à Eletricidade Estática

Todos os dispositivos eletrônicos, em diversos graus, podem ser danificados pela eletricidade estática. O corpo humano é um grande armazém de cargas eletrostáticas. Por exemplo, quando andamos sobre um carpete, uma carga estática correspondente a mais de 30.000 V pode ser armazenada em nosso corpo. Se você tocar um dispositivo eletrônico, uma parte desta carga considerável pode ser transferida para o dispositivo. As famílias lógicas MOS (e todos os MOSFETs) são especialmente susceptíveis a dano por eletricidade estática. Toda essa diferença de potencial (carga eletrostática) aplicada pela fina camada de óxido

supera a capacidade de isolamento elétrico desta camada. Quando esta se rompe, o fluxo de corrente resultante (descarga) é semelhante à queda de um raio, e faz um furo na camada de óxido, danificando permanentemente o dispositivo.

A **descarga eletrostática (ESD —** *electrostatic discharge*) é responsável por bilhões de dólares anuais em danos em equipamentos eletrônicos, e os fabricantes de equipamentos têm dedicado considerável atenção a desenvolver procedimentos especiais de manuseio, para todos os dispositivos e circuitos eletrônicos. Mesmo que os CIs mais modernos tenham uma rede resistor-diodo interna para proteger entradas e saídas dos efeitos da ESD, as seguintes precauções* são usadas pela maioria dos laboratórios de engenharia, unidades de produção e departamentos de serviço em campo:

1. Conecte o chassi de todos os instrumentos de teste, pontas de ferros de solda e sua bancada de trabalho (se ela for de metal) ao terra da rede (isto é, ao pino redondo da tomada de 120 VAC). Isto impede o acúmulo de carga estática nestes dispositivos, que poderia ser transferida para qualquer placa de circuito impresso ou CI que entrasse em contato com os mesmos.
2. Conecte-se ao terra da rede com uma pulseira especial. Isto vai permitir que as cargas potencialmente perigosas do seu corpo sejam descarregadas para a terra. A pulseira especial contém um resistor de 1 MΩ, que limita a corrente a um valor não-letal para o caso de você acidentalmente tocar numa tensão "viva" enquanto estiver trabalhando com o equipamento.
3. Mantenha os CIs (especialmente os MOS) em espuma condutora ou sobre folha de alumínio. Isto vai fazer com que todos os pinos estejam em curto, impedindo que diferenças de potencial perigosas possam se desenvolver entre dois pinos.
4. Evite tocar os pinos dos CIs, e insira-os imediatamente no circuito após removê-los da embalagem protetora.
5. Coloque estrapes de curto nos conectores de borda de placas de circuito impresso quando estas estiverem sendo transportadas. Evite tocar os conectores de borda. Armazene placas de circuito impresso em plástico condutor ou envelopes metálicos.
6. Não deixe as entradas não-utilizadas dos CIs em aberto, porque elas tendem a acumular eletricidade estática.

Questões de Revisão

1. Descreva as vantagens e desvantagens dos circuitos N-MOS quando comparados aos TTL.
2. O que causa as limitações de fan-out da lógica MOS?
3. Que fatores limitam a velocidade de comutação dos circuitos N-MOS?
4. O que torna os circuitos MOS indicados para aplicações LSI?
5. Por que CIs MOS são especialmente sensíveis à eletricidade estática?
6. Enumere cinco precauções para o manuseio de CIs e placas de circuitos para minimizar danos devidos à eletricidade estática.

*As precauções 1 e 2 só se aplicam em instalações que possuem os três pinos (fase, neutro e terra) corretamente ligados. Note que isto não é comum nas instalações residenciais no Brasil, onde apenas dois pinos (fase e neutro) são utilizados (*N. do T.*)

8-14 LÓGICA MOS COMPLEMENTAR

A família lógica *MOS complementar* **(CMOS)** utiliza MOSFETs *tanto* de canal-P *quanto* de canal-N para obter diversas vantagens sobre as famílias N-MOS e P-MOS. De um modo geral, CMOS é mais rápido e consome ainda menos que as outras famílias MOS. Estas vantagens são contrabalançadas pelo aumento de complexidade para a fabricação do CI e pela menor densidade de integração.

INVERSOR CMOS

O circuito básico de INVERSOR CMOS é mostrado na Fig. 8-35. Neste diagrama e para outros que se seguirão, os símbolos padronizados para MOSFETs foram trocados por blocos com as denominações P e N para indicar um MOSFET-P e um MOSFET-N, respectivamente. Isto é feito por conveniência na análise dos circuitos. O INVERSOR CMOS tem dois MOSFETs em série, de modo que o dispositivo com canal P tem sua fonte conectada a $+V_{DD}$ (uma tensão positiva), e o dispositivo de canal N tem sua fonte conectada à terra.* As portas dos dois dispositivos estão conectadas juntas em uma entrada comum. Os drenos dos dois dispositivos estão conectados juntos em uma saída comum.

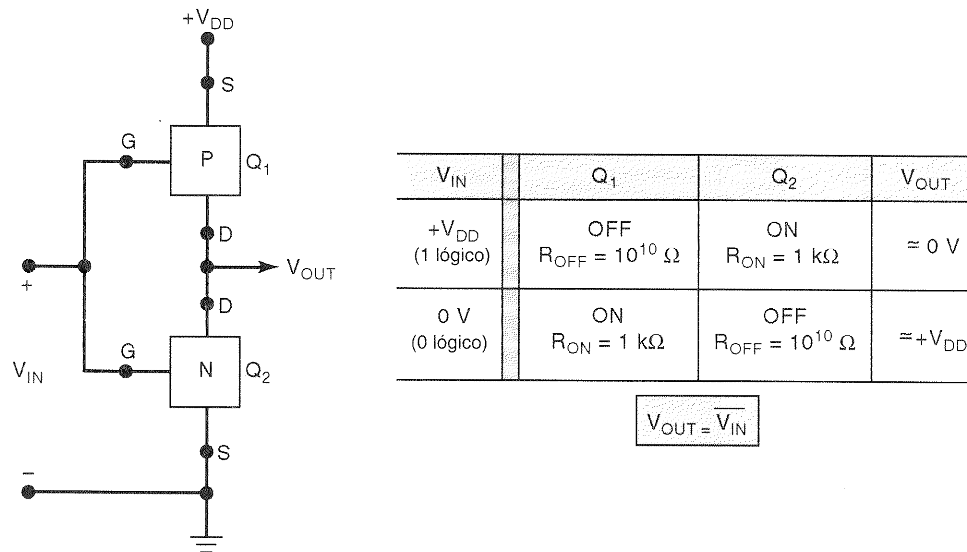
Os níveis lógicos CMOS são essencialmente $+V_{DD}$, para o 1 lógico, e 0 V, para o 0 lógico. Considere primeiro o caso em que $V_{IN} = +V_{DD}$. Nesta situação, a porta de Q_1 (canal P) está em 0 V em relação à fonte de Q_1. Então, Q_1 estará no seu estado OFF com $R_{OFF} \approx 10^{10}\ \Omega$. A porta de Q_2 (canal N) estará com $+V_{DD}$ em relação à sua fonte. Portanto, Q_2 estará, tipicamente, com $R_{ON} = 1\ k\Omega$. O divisor de tensão entre R_{OFF} de Q_1 e R_{ON} de Q_2 produzirá $V_{OUT} \approx 0$ V.

Agora, considere o caso em que $V_{IN} = 0$ V. Q_1 agora tem sua porta em um potencial negativo em relação à sua fonte, enquanto Q_2 tem $V_{GS} = 0$ V. Portanto, Q_1 estará ligado com $R_{ON} = 1\ k\Omega$ e Q_2 desligado com $R_{OFF} = 10^{10}\ \Omega$, produzindo um V_{OUT} de aproximadamente $+V_{DD}$. Esses dois estados de operação estão resumidos na Fig. 8-35(b), mostrando que o circuito age como um INVERSOR lógico.

Porta NAND CMOS

Outras funções lógicas podem ser construídas modificando-se o INVERSOR básico. A Fig. 8-36 mostra uma porta NAND formada pela adição de um MOSFET canal-P, em paralelo, e um MOSFET canal-N, em série ao INVERSOR básico. Para analisar este circuito, é importante perceber que uma entrada em 0 V liga seu MOSFET-P correspondente e desliga seu MOSFET-N correspondente. O oposto ocorre para uma entrada em $+V_{DD}$. Portanto, podemos observar

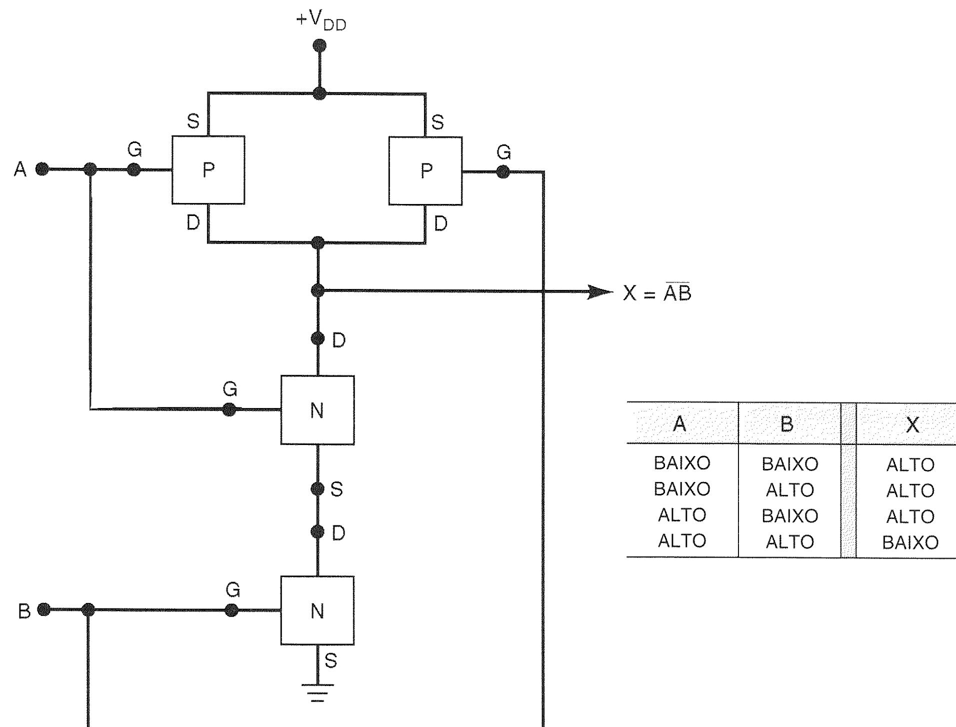
*A maioria dos fabricantes denomina este terminal V_{SS}.



V_{IN}	Q_1	Q_2	V_{OUT}
$+V_{DD}$ (1 lógico)	OFF $R_{OFF} = 10^{10}\ \Omega$	ON $R_{ON} = 1\ k\Omega$	≈ 0 V
0 V (0 lógico)	ON $R_{ON} = 1\ k\Omega$	OFF $R_{OFF} = 10^{10}\ \Omega$	$\approx +V_{DD}$

$V_{OUT} = \overline{V_{IN}}$

Fig. 8-35 - INVERSOR CMOS BÁSICO.



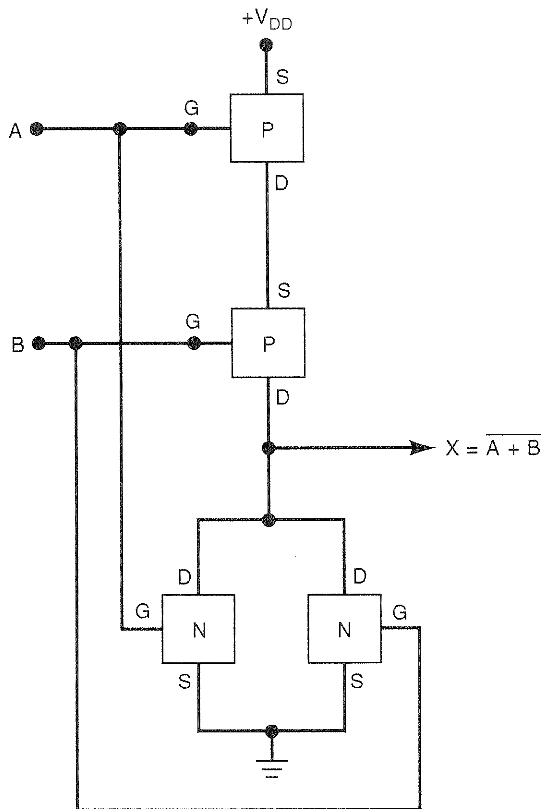
A	B	X
BAIXO	BAIXO	ALTO
BAIXO	ALTO	ALTO
ALTO	BAIXO	ALTO
ALTO	ALTO	BAIXO

Fig. 8-36 - Porta NAND CMOS.

que o único instante em que uma saída em BAIXO ocorrerá será quando as entradas *A* e *B* estiverem ambas em ALTO ($+ V_{DD}$) para ligar ambos os MOSFETs canal-N, fornecendo assim uma resistência baixa entre o terminal de saída e a terra. Para todas as outras condições de entrada, pelo menos um MOSFET-P estará ligado, enquanto pelo menos um MOSFET-N estará desligado. Isto produz uma saída em ALTO.

Porta NOR CMOS

A porta NOR CMOS é formada adicionando um MOSFET-P em série e um MOSFET-N em paralelo ao INVERSOR básico, como mostrado na Fig. 8-37. Mais uma vez, este circuito pode ser analisado, observando que um nível baixo em qualquer uma das entradas liga o seu MOSFET-P correspondente e desliga o seu MOSFET-N correspondente, e o oposto



A	B	X
BAIXO	BAIXO	ALTO
BAIXO	ALTO	BAIXO
ALTO	BAIXO	BAIXO
ALTO	ALTO	BAIXO

Fig. 8-37 - Porta NOR CMOS.

ocorre para uma entrada em ALTO. Cabe ao leitor verificar que este circuito opera com uma porta NOR.

Portas AND e OR podem ser formadas através da combinação de NANDs e NORs com INVERSORes.

FLIP-FLOP SET-CLEAR CMOS

Duas portas NOR ou NAND CMOS podem ser ligadas com acoplamento cruzado para formar um simples latch SET-CLEAR. Portas adicionais são usadas para converter um latch SET-CLEAR básico em flip-flops D e J-K com clock.

Questões de Revisão

1. De que forma o circuito interno CMOS difere do N-MOS?
2. Quantos MOSFETs canal-P existem em um INVERSOR CMOS?
3. Quantos MOSFETs existem numa porta NAND de três entradas?

8-15 CARACTERÍSTICAS DA SÉRIE CMOS

A família CMOS de circuitos integrados compete diretamente com a TTL nas áreas de integração em pequena e média escalas (SSI, MSI). Como a tecnologia CMOS tem produzido uma performance cada vez melhor, CMOS vem gradualmente tomando terreno que tem sido há muito dominado pela tecnologia TTL. Componentes TTL estarão presentes ainda por muito tempo, porém cada vez mais novos equipamentos estão utilizando circuitos lógicos predominantemente CMOS.

CIs CMOS fornecem não apenas as mesmas funções lógicas disponíveis na família TTL mas também várias funções especiais que não estão disponíveis na TTL. Várias séries CMOS diferentes foram desenvolvidas ao longo do tempo, à medida que os fabricantes procuravam melhorar as características de performance. Antes de estudarmos as diversas séries CMOS, é importante definir alguns termos, que serão usados quando CIs de famílias ou séries diferentes forem usados juntos ou como substitutos de uma para outra.

- **Compatível pino a pino**. Dois CIs são compatíveis pino a pino quando suas pinagens são as mesmas. Por exemplo, o pino 7 em ambos os CIs é TERRA, o pino 1 em ambos é a entrada do primeiro INVERSOR, e assim por diante.
- **Funcionalmente equivalente**. Dois CIs são funcionalmente equivalentes quando as funções lógicas que eles realizam são exatamente as mesmas. Por exemplo, ambos contêm quatro portas NAND de duas entradas ou ambos contêm seis flip-flops do tipo D disparados pela transição positiva do sinal de clock.
- **Eletricamente compatível**. Dois CIs são eletricamente compatíveis quando eles podem ser ligados diretamente um ao outro, sem a necessidade de se tomarem precauções especiais para assegurar a operação correta.

Série 4000/14000

A mais antiga série CMOS é a série 4000, que foi primeiramente produzida pela RCA e é funcionalmente equivalente à série 14000 da Motorola. Os componentes da série 4000/14000 têm um consumo muito baixo e podem operar em uma larga faixa de tensões de alimentação (3 a 15 V). Eles são muito lentos quando comparados com TTL ou com as outras séries CMOS e possuem uma capacidade de corrente de saída muito baixa. Não são compatíveis nem pino a pino nem eletricamente a nenhuma das séries TTL. Os componentes da série 4000/14000 raramente são utilizados em novos projetos, exceto quando um CI de uso especial desta série é necessário e não está disponível em nenhuma outra.

Série 74C

Esta série CMOS é pino a pino compatível e funcionalmente equivalente a componentes TTL com a mesma numeração. Por exemplo, um 74C74 é um duplo flip-flop D disparado pela borda, que tem a mesma pinagem que o TTL 7474, que também é duplo flip-flop D disparado pela borda. Muitas das funções, mas não todas, que estão disponíveis para TTL também estão disponíveis nesta série CMOS. Isto torna possível substituir alguns circuitos TTL por um equivalente CMOS. As características de performance da série 74C são quase as mesmas da série 4000.

74HC/HCT (High Speed CMOS — CMOS de Alta Velocidade)

Esta é uma versão aperfeiçoada da série 74C, que tem um aumento na velocidade de operação, agora comparável a componentes 74LS, e uma capacidade de corrente muito mais alta do que a da série 74C. Os CIs 74HC/HCT são compatíveis pino a pino e funcionalmente equivalentes a CIs TTL com a mesma numeração. Os componentes 74HCT são eletricamente compatíveis com TTL, mas os 74HC não são. Isto significa, por exemplo, que o chip INVERSOR sêxtuplo 74HCT04 pode substituir um chip 74LS04, e vice-versa. Isto também significa que um CI 74HCT pode ser conectado diretamente com qualquer CI TTL. A série 74HC/HCT tem se tornado a mais amplamente utilizada das séries CMOS.

74AC/ACT (CMOS Avançado)

Freqüentemente fazemos referência a esta série como ACL (*advanced cmos logic* — lógica CMOS avançada). Esta série é funcionalmente equivalente a várias séries TTL, mas *não* é compatível pino a pino com TTL. A razão para isto é que a localização dos pinos nos chips, 74AC ou 74ACT, foi escolhida para melhorar a imunidade ao ruído, de modo que as entradas destes componentes sejam menos sensíveis às transições de sinais que ocorram em outros pinos do CI. Os componentes 74AC não são eletricamente compatíveis com TTL. Os componentes 74ACT podem ser conectados diretamente a componentes TTL. A série ACL oferece vantagens sobre as séries HC no que se refere a imunidade a ruído, atraso de propagação e máxima freqüência de clock.

A numeração dos componentes desta série difere um pouco da numeração TTL, 74C e 74HC/HCT. Ela utiliza um número de componente com cinco dígitos, começando com os dígitos 11. Os exemplos a seguir ilustram este fato.

74AC11 004 ≡ 74HC 04
74ACT11 293 ≡ 74HCT 293

74AHC (Advanced High-Speed CMOS — CMOS Avançado de Alta Velocidade)

Esta é a mais nova série de dispositivos CMOS, e oferece uma migração natural das séries HC para aplicações de mais alta velocidade, baixo consumo e baixa capacidade de acionamento. Os componentes desta série são três vezes mais rápidos e podem ser usados como substitutos diretos de componentes da série HC. Eles oferecem imunidade ao ruído semelhante à da série HC, sem os problemas de transientes de chaveamento geralmente associados com as características de acionamento necessárias para esta velocidade.

Lógica BiCMOS

Vários fabricantes de circuitos integrados desenvolveram séries que combinam as melhores características da lógica bipolar e da CMOS. O baixo consumo do CMOS e a alta velocidade dos circuitos bipolares são integrados para produzir uma família lógica de consumo extremamente baixo e de alta velocidade. A maioria das funções SSI e MSI não está disponível em CIs BiCMOS; estes estão limitados a funções que são usadas para interface com microprocessadores e em aplicações que envolvam memória, como latches, buffers, drivers e transceptores. A série 74BCT (BiCMOS Bus-Interface Technology — Tecnologia de Interface de Barramento) oferece uma redução de 75% no consumo em relação à família 74F, ao mesmo tempo que mantém a velocidade de operação e as características de acionamento similares. Os componentes desta série são compatíveis pino a pino com os componentes TTL padrão e operam com níveis lógicos segundo o padrão de 5 V. A série 74ABT (Advanced BiCMOS Technology — Tecnologia BiCMOS Avançada) é a segunda geração dos dispositivos de interface de barramento BiCMOS. Uma vez que as velocidades em sistemas computacionais vêm aumentando, esta família oferece ao projetista de sistemas componentes que parecerão transparentes aos sinais (isto é, não terão efeitos negativos sobre estes) enquanto estiverem conectando componentes de alta velocidade em um computador. Alguns destes componentes desta série são projetados para sistemas de 3,3 V, que serão descritos depois em uma outra seção.

Tensão de Alimentação

As séries 4000/14000 e 74C vão operar com V_{DD} variando de 3 a 15 V, o que torna estes circuitos muito versáteis. Eles podem ser usados em circuitos alimentados com bateria, em circuitos com alimentação padrão de 5 V e em circuitos onde uma tensão de alimentação maior é usada para atender às margens de ruído necessárias para o funcionamento do circuito em ambiente ruidoso. As séries 74HC/HCT e 74AC/ACT operam com tensões de alimentação em uma faixa muito mais estreita, geralmente entre 2 e 6 V.

Sempre que componentes CMOS e TTL são utilizados juntos, a tensão de alimentação é geralmente igual a 5 V, de modo que uma única fonte de alimentação de 5 V pode servir tanto como V_{DD}, para os circuitos CMOS, quanto para V_{CC}, para os TTL. Em outras situações em que dispositivos CMOS devem operar com uma fonte de alimentação diferente de 5 V, medidas especiais devem ser tomadas se componentes CMOS e TTL forem conectados. Examinaremos estas medidas mais adiante.

Níveis de Tensão Lógicos

Os níveis de tensão de entrada e saída são diferentes para cada uma das séries CMOS. A Tabela 8-10 relaciona os valores de tensão para as várias séries CMOS, bem como para as séries TTL. Os valores relacionados assumem que todos os dispositivos estão operando com uma tensão de alimentação igual a 5 V e que todos os dispositivos estão acionando entradas da mesma família lógica.

O exame dessa tabela mostra alguns pontos importantes. Primeiro, note que V_{OL} para dispositivos CMOS é muito próximo de 0 V, e V_{OH} é muito próximo de 5 V. A razão disto é que as saídas CMOS não têm que fornecer ou absorver uma quantidade significativa de corrente quando estão acionando entradas CMOS porque estas possuem resistência de entrada muito alta (10^{12} Ω). Observe também que, exceto para as famílias 74HCT e 74ACT, os níveis de tensão de entrada necessários são maiores para CMOS do que para TTL. Lembre-se de que as séries 74HCT e 74ACT foram projetadas para serem eletricamente compatíveis com TTL, e portanto devem ser capazes de aceitar os mesmos níveis de tensão de entrada das séries TTL.

Margens de Ruído

As margens de ruído para cada família também são dadas na Tabela 8-10. Elas são calculadas usando

$$V_{NH} = V_{OH}(\text{min}) - V_{IH}(\text{min})$$
$$V_{NL} = V_{IL}(\text{max}) - V_{OL}(\text{max})$$

Observe que, de um modo geral, os dispositivos CMOS têm margens de ruído maiores que os TTL. A diferença seria ainda maior se os dispositivos CMOS estivessem operando com uma fonte de alimentação maior do que 5 V.

Dissipação de Potência

Quando o circuito lógico CMOS está estático (não está comutando), sua dissipação de potência é extremamente baixa. Podemos ver a razão disto pelo exame de cada um dos circuitos mostrados nas Fig. 8-35 a 8-37. Note que, independentemente do estado da saída, existe sempre uma resistência muito alta entre os terminais V_{DD} e terra, porque existe sempre um MOSFET desligado no caminho da corrente. Em resultado, a dissipação de potência DC típica dos circuitos CMOS é de apenas 2,5 nW por porta quando V_{DD} = 5 V. Mesmo quando V_{DD} = 10 V esta dissipação de potência aumenta para apenas 10 nW. Com estes valores de dissipação de potência, é fácil entender por que dispositivos CMOS são especialmente indicados para aplicações em que os circuitos são alimentados por bateria, ou em que existe um sistema de emergência com bateria.

Dissipação de Potência Aumenta com a Freqüência

A dissipação de potência de um CI CMOS será bastante baixa desde que ele esteja em uma condição DC, isto é, com a saída em um nível constante. Infelizmente, a dissipação de potência aumentará proporcionalmente à freqüência de comutação de estados do circuito. Por exemplo, uma porta NAND CMOS dissipa 10 nW quando está em uma das condições DC, e dissipa 0,1 mW em uma freqüência de 100 kpps e 1 mW a 1 MHz. A razão para esta dependência em relação à freqüência está ilustrada na Fig. 8-38.

Cada vez que uma saída CMOS comuta de BAIXO para ALTO, uma corrente transiente deve ser fornecida para a capacitância de carga. Esta capacitância consiste na combinação de todas as capacitâncias de entrada de quaisquer cargas que estiverem sendo acionadas com a capacitância da saída do dispositivo. Estes pulsos estreitos de corrente devem ser fornecidos por V_{DD} e podem ter uma amplitude típica de 5 mA com uma duração de 20 a 30 ns. Obviamente, à medida que a freqüência de comutação aumenta, existirá um maior número destes pulsos de corrente ocorrendo por segundo, e, portanto, a corrente média drenada de V_{DD} vai aumentar. Mesmo com capacitâncias de carga muito baixas, existe um breve momento da transição de BAIXO para ALTO ou de ALTO para BAIXO em que os dois transistores de saída estão conduzindo parcialmente. Isto efetivamente diminui a resistência entre a fonte de alimentação e a terra, causando também um pulso de corrente.

Então, em freqüências mais altas, as séries CMOS começam a perder algumas de suas vantagens sobre outras famílias lógicas. Como uma regra geral, uma porta CMOS terá a mesma dissipação média que uma porta 74LS em freqüências próximas de 2 a 3 MHz. Para chips MSI, a situação é mais complexa do que aquela exposta aqui, e o projetista

TABELA 8-10 Níveis de tensão (em volts) de entrada/saída com $V_{DD} = V_{CC}$ = + 5 V.

	CMOS							TTL			
Parâmetro	**4000B**	**74HC**	**74HCT**	**74AC**	**74ACT**	**74AHC**	**74AHCT**	**74**	**74LS**	**74AS**	**74ALS**
V_{IH}(min)	3,5	3,5	2,0	3,5	2,0	3,85	2,0	2,0	2,0	2,0	2,0
V_{IL}(max)	1,5	1,0	0,8	1,5	0,8	1,65	0,8	0,8	0,8	0,8	0,8
V_{OH}(min)	4,95	4,9	4,9	4,9	4,9	4,4	3,15	2,4	2,7	2,7	2,7
V_{OL}(max)	0,05	0,1	0,1	0,1	0,1	0,44	0,1	0,4	0,5	0,5	0,4
V_{NH}	1,45	1,4	2,9	1,4	2,9	0,55	1,15	0,4	0,7	0,7	0,7
V_{NL}	1,45	0,9	0,7	1,4	0,7	1,21	0,7	0,4	0,3	0,3	0,4

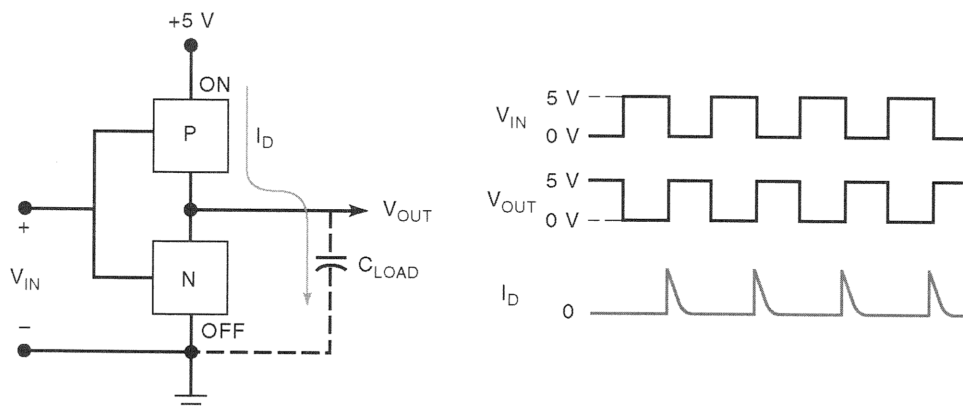


Fig. 8-38 - Spikes de corrente são drenados da fonte V_{DD} toda vez que a saída comuta de BAIXO para ALTO. Isto é devido principalmente à corrente necessária para carregar a carga capacitiva.

deve fazer uma análise detalhada, para determinar se a série CMOS em questão apresenta alguma vantagem em relação à dissipação de potência na freqüência de operação.

Fan-Out

Do mesmo modo que as entradas N-MOS e P-MOS, as CMOS têm uma resistência extremamente alta (10^{12} Ω) e drenam quase nenhuma corrente da fonte de sinal. Cada entrada CMOS, entretanto, geralmente apresenta uma carga de 5 pF para terra. Esta capacitância de entrada limita o número de entradas CMOS que uma saída CMOS pode acionar (veja Fig. 8-39). A saída CMOS deve carregar e descarregar a combinação de todas as capacitâncias de entrada em paralelo, de modo que o tempo de comutação da saída aumentará proporcionalmente ao número de cargas que estiverem sendo acionadas. Tipicamente, cada carga CMOS aumenta o atraso de propagação do circuito acionador em 3 ns. Por exemplo, a porta NAND 1 na Fig. 8-39 teria um t_{PLH} de 25 ns se não estivesse acionando carga nenhuma. Este valor aumentaria para 25 ns + 20(3 ns) = 85 ns, se ela estivesse acionando *vinte* cargas.

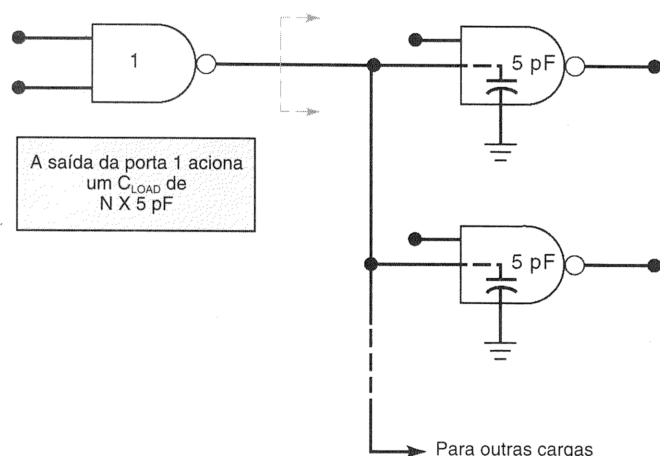


Fig. 8-39 - Cada entrada CMOS contribui para a capacitância de carga total da saída da porta acionadora.

Portanto, o fan-out CMOS depende do atraso de propagação máximo permitido. Geralmente, as saídas CMOS estão limitadas a um fan-out de 50 para operação em baixa freqüência (≤ 1 MHz). É claro que para operação em alta freqüência o fan-out teria que ser diminuído.

Velocidade de Comutação

Apesar de circuitos CMOS, do mesmo modo que os N-MOS ou P-MOS, terem que acionar cargas capacitivas relativamente grandes, sua velocidade de comutação é relativamente rápida devido à baixa resistência de saída em cada estado. Lembre-se de que uma saída N-MOS deve carregar uma carga capacitiva através de um resistor relativamente grande (100 kΩ). No circuito CMOS, a resistência de saída no estado ALTO é a R_{ON} do MOSFET-P, que é em geral menor ou igual a 1 kΩ. Isto permite uma carga mais rápida do capacitor de carga.

Uma porta NAND da série 4000 terá tipicamente um t_{pd} de 50 ns com V_{DD} = 5 V, e 25 ns com V_{DD} = 10 V. A razão para esta melhora no t_{pd} quando V_{DD} aumenta é que a R_{ON} do MOSFET diminui significativamente quando ele é alimentado com tensões mais altas. Então, pode parecer que V_{DD} deveria ser o maior possível para que o circuito pudesse operar em altas freqüências. Entretanto, um V_{DD} maior resultaria em um aumento de dissipação de potência.

Uma típica porta NAND da série 74HC/HCT tem um t_{pd} médio em torno de 8 ns quando V_{DD} = 5 V. Uma porta NAND 74AC/ACT tem um t_{pd} médio em torno de 4,7 ns. Uma porta NAND 74AHC tem um t_{pd} médio em torno de 4,3 ns.

Entradas Não-utilizadas

Entradas CMOS nunca devem ficar desconectadas. Todas as entradas CMOS devem ser conectadas a um nível de tensão fixo (0 V ou V_{DD}) ou a alguma outra entrada.

Esta regra se aplica também a entradas de portas lógicas que não foram usadas em um chip. Uma entrada CMOS não-conectada é susceptível a ruído e a eletricidade estática, que poderiam facilmente polarizar os MOSFETs canal-P e canal-

N para um estado de condução, resultando no aumento da dissipação de potência e em possível superaquecimento.

Sensibilidade à Eletricidade Estática

A grande resistência das entradas CMOS as torna especialmente sensíveis ao acúmulo de cargas estáticas, que podem produzir tensões suficientemente grandes para romper a isolação dielétrica entre a porta e o canal do MOSFET. A maioria dos dispositivos CMOS mais modernos é protegida, contra danos causados por eletricidade estática, por redes diodo-resistor em cada entrada. Estes diodos são projetados para entrar em condução e limitar a tensão de entrada, bem abaixo de um valor que possa causar algum estrago. Ainda que, na maioria dos casos, estes diodos sejam eficientes, às vezes, eles não começam a conduzir com a rapidez suficiente para evitar que o CI seja danificado; por isso, ainda é uma boa idéia observar os cuidados especiais de manuseio vistos na Seção 8-13.

Latch-Up

Devido à presença inevitável de transistores PNP e NPN *parasitas* (indesejados) no substrato dos CIs CMOS, uma condição conhecida como **latch-up** pode ocorrer em certas circunstâncias. Se estes transistores parasitas em um chip CMOS são disparados para condução, eles vão ficar travados (permanentemente ligados), e uma grande corrente poderá destruir o CI. A maioria dos CIs CMOS mais modernos possui um circuito de proteção que ajuda a prevenir o latch-up. Entretanto, isto ainda pode ocorrer quando os valores máximos de tensão forem excedidos. Latch-up pode acontecer devido a spikes de alta tensão ou devido às oscilações nas entradas e saídas do componente. Diodos grampeadores podem ser colocados externamente para proteger os CIs destes transientes, especialmente quando estes são usados em ambientes industriais, onde a comutação de alta tensão e/ou de grandes cargas de corrente acontece (controladores de motores, relés etc.). Uma fonte de alimentação bem-regulada vai minimizar estes spikes em V_{DD}. Se a fonte também possuir um limitador de corrente, ela vai limitar a corrente antes que o latch-up ocorra. Além disso, todas as entradas CMOS não-utilizadas devem ser conectadas a TERRA ou V_{DD} para reduzir a chance de transientes de tensão espúrios aparecerem nestas entradas e causarem o latch-up.

Comparação entre as Séries CMOS e TTL

A Tabela 8-11 compara os valores típicos de alguns parâmetros das principais séries de CIs digitais.

Questões de Revisão

1. Quais séries CMOS são compatíveis pino a pino com as TTL?
2. Quais séries CMOS são eletricamente compatíveis com as TTL?
3. Quais séries CMOS são funcionalmente equivalentes às TTL?
4. Que família lógica combina as melhores características da lógica CMOS e da bipolar?
5. Que fatores determinam o fan-out de dispositivos CMOS?
6. Que precauções devem ser tomadas quando se manuseiam CIs CMOS?
7. Que família de CIs é mais indicada para aplicações alimentadas por bateria?
8. *Verdadeiro ou falso:*
   (a) O consumo de potência aumenta com a freqüência de operação para componentes CMOS.
   (b) Entradas CMOS não-utilizadas podem ser deixadas desconectadas.
   (c) Componentes TTL são mais indicados que os CMOS para operação em ambientes ruidosos.
   (d) A velocidade de comutação no componente CMOS aumenta com a freqüência.
   (e) A velocidade de comutação no componente CMOS aumenta com a fonte de alimentação
   (f) A condição de latch-up é uma vantagem dos componentes CMOS sobre os TTL.

8-16 TECNOLOGIA DE BAIXA TENSÃO

Os fabricantes de circuitos integrados estão sempre procurando maneiras de colocar um número maior de dispositivos semicondutores (diodos, transistores, resistores etc.) juntos em um chip, isto é, procuram aumentar a densidade de integração. Esta densidade mais alta traz dois grandes

TABELA 8-11 Comparação entre as séries de CIs digitais.*

	4000B	74HC/HCT	74AC/ACT	74AHC/T	74	74LS	74AS	74ALS	ECL
Dissipação de potência por porta (mW)									
Estática	$1{,}0 \times 10^{-3}$	$2{,}5 \times 10^{-3}$	$5{,}0 \times 10^{-3}$	$9{,}0 \times 10^{-5}$	10	2	8	1,2	25
A 100 kHz	0,1	0,17	0,08	0,006	10	2	8	1,2	25
Atraso de propagação (ns)	50	8	4,7	3,7	9	9,5	1,7	4	1
Velocidade-potência (a 100 kHz) (pJ)	5	1,4	0,37	0,02	90	19	13,6	4,8	25
Freqüência máxima de clock (MHz)	12	40	100	130	35	45	200	70	300
Margem de ruído no pior caso (V)	1,5	0,9	0,7	0,55	0,4	0,3	0,3	0,4	0,25

*Todos os valores para CMOS são dados para V_{DD} = 5 V. Parâmetros para dispositivos específicos podem variar.

benefícios: primeiro, permite que mais circuitos sejam colocados no chip, e segundo, com os circuitos colocados mais próximos uns dos outros, o tempo de propagação de um circuito para outro diminui, e, portanto, aumenta a velocidade de operação dos circuitos como um todo. Também existem desvantagens quando se tem chips com densidades maiores. Quando circuitos são colocados muito próximos uns dos outros, o material isolante entre eles é mais estreito. Isto diminui o valor de tensão que o dispositivo pode suportar antes que o dielétrico se rompa. Aumentar a densidade do chip aumenta também o consumo de potência, o que pode fazer com que a temperatura do chip aumente para um valor acima do permitido para uma operação confiável.

Estas desvantagens podem ser neutralizadas fazendo-se com que o chip funcione com um nível de tensão mais baixo. Fabricantes de circuitos integrados estão fazendo isto, desenvolvendo uma nova linha de dispositivos lógicos que opera com uma tensão de alimentação nominal de 3,3 V, em vez do tradicional 5 V. Alguns padrões foram adotados desde 1984 para definir as características de tensão e corrente destes dispositivos de baixa tensão. Entretanto, à medida que as aplicações vão se desenvolvendo, os requisitos têm mudado, e por isso novos padrões estão sendo desenvolvidos. Esta tecnologia de baixa tensão pode ser o início de uma transição gradual nos equipamentos digitais que eventualmente terão todos os seus CIs operando no padrão 3,3 V.

Dispositivos de baixa tensão são atualmente projetados para aplicações que vão desde jogos eletrônicos a estações de trabalho de engenharia. Os microprocessadores mais novos são dispositivos de 3,3 V, e chips de memória dinâmica de 64 bits e 3,3 V estão disponíveis em módulos de memória para computadores. A operação em baixa tensão é particularmente útil em equipamentos alimentados por bateria, onde o número de células necessárias é reduzido.

Várias famílias lógicas de 3,3 V estão disponíveis atualmente. Não é possível abordar todas as séries de todos os fabricantes, então descreveremos aquelas atualmente oferecidas pela Texas Instruments.

- A série *74LVC* (*Low-Voltage CMOS — CMOS de Baixa Tensão*) contém a maior coleção de portas SSI e funções MSI das famílias de 5 V, juntamente com vários dispositivos de interface de barramento, tais como buffers, latches, drivers etc. Esta série é capaz de lidar com níveis lógicos de 5 V em suas entradas, então, é capaz de fazer a conversão de níveis lógicos de sistemas de 5 V para sistemas de 3,3 V. Desde que a corrente seja mantida suficientemente baixa para manter a tensão de saída em níveis aceitáveis, a 74LVC também pode acionar entradas TTL de 5 V. Os requisitos de entrada V_{IH} da série CMOS de 5 V como a 74HC/AHC não permitem que dispositivos LVC os acionem.
- A série *74ALVC* (*Advanced Low Voltage CMOS — CMOS de Baixa Tensão Avançado*) oferece a melhor performance. Os dispositivos desta série são destinados principalmente a aplicações de interface de barramento que utilizam apenas lógica de 3,3 V.
- A série *74LV* (*Low Voltage — Baixa Tensão*) oferece tecnologia CMOS e muitas das portas SSI e funções MSI comuns, juntamente com alguns buffers octais, latches e flip-flops mais populares. Foi projetada para operar somente com outros dispositivos de 3,3 V.
- A série *74LVT* (*Low Voltage BiCMOS Technology — Tecnologia BiCMOS de Baixa Tensão*) contém dispositivos BiCMOS que foram projetados para aplicações de interface de barramentos de 8 e 16 bits. Do mesmo modo que a série LVC, as entradas podem lidar com níveis lógicos de 5 V e servir como um conversor de 5 V para 3 V. Uma vez que os níveis de saída [V_{OH}(min) e V_{OL}(max)] são equivalentes a níveis TTL, eles são eletricamente compatíveis com TTL. A Tabela 8-12 compara as diversas características.

Sem dúvida, outras famílias serão introduzidas nesta fase de transição. Os engenheiros e técnicos não podem pressupor que todo CI em um circuito digital esteja operando com 5 V, e devem estar preparados para lidar com as considerações necessárias de interface em sistemas com várias tensões de alimentação. A capacidade de interfacear que você aprenderá neste capítulo permitirá que você consiga lidar com estas considerações, independentemente do que estiver sendo desenvolvido, à medida que nos aproximamos do ano 2000.

O desenvolvimento contínuo da LVT promete trazer um revolução completa dos atuais sistemas de 5 V para sistemas com tensão de alimentação mista e, finalmente, para projetos somente com lógica de 3,3 V. Embora a lógica de 5 V ainda vá permanecer conosco por um bom tempo, seus dias parecem estar contados. Para que você tenha uma idéia disto, a Fig. 8-40 mostra a percepção da Texas Instruments do ciclo de vida das várias famílias lógicas.

Questões de Revisão

1. Quais são as duas vantagens de CIs de alta densidade?
2. Quais são as desvantagens?
3. Qual é o valor mínimo para entrada 74LVT em ALTO?
4. Quais são as séries de baixa tensão que podem operar apenas com outros CIs de séries de baixa tensão?
5. Quais são as séries de baixa tensão que são completamente compatíveis eletricamente com TTL?

TABELA 8-12 Características das séries de baixa tensão.

	LVC	ALVC	LV	LVT
V_{CC}(recomendado)	2,0 a 3,6	2,3 a 3,6	2,7 a 3,6	2,7 a 3,6
t_{pd} (ns)	6,5 ns	3 ns	18 ns	4 ns
Intervalo para $V_{IH}^{(V)}$	2,0 a 6,5	2,0 a 4,6	2,0 a V_{CC} + 0,5	2,0 a 7
V_{IL}(max) (V)	0,8	0,8	0,8	0,8
I_{OH}(mA)	24	12	6	32 mA
I_{OL}(mA)	24	12	6	64 mA

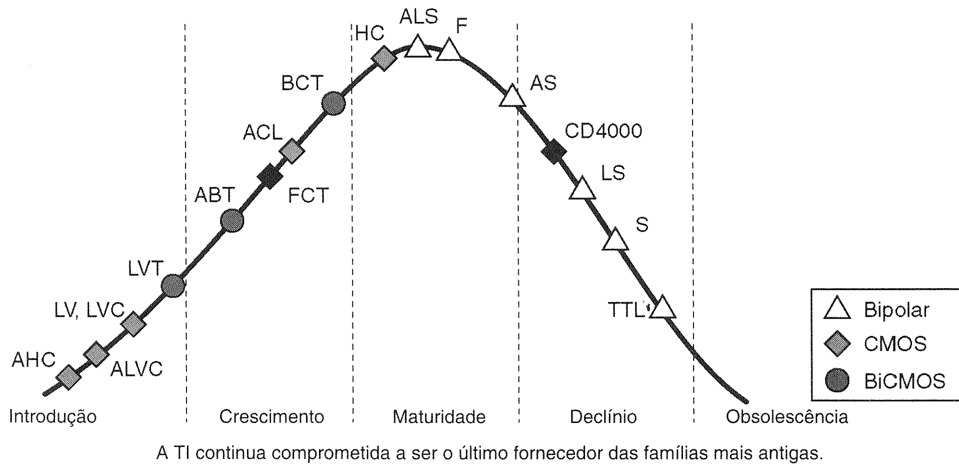


Fig. 8-40 - Ciclo de vida das famílias lógicas.

8-17 SAÍDAS CMOS DE DRENO ABERTO E TRISTATE

Saídas CMOS convencionais nunca devem ser conectadas juntas.

A Fig. 8-41 mostra o que pode acontecer quando as saídas de dois INVERSORes CMOS são colocadas em curto. A saída da esquerda está tentando ir para o estado ALTO, e seu MOSFET-P está conduzindo com R_{ON} = 1 kΩ. A outra entrada está tentando ir para o estado BAIXO, e seu MOSFET-N está conduzindo com R_{ON} = 1 kΩ. O terminal de saída comum *X* estará com uma tensão de aproximadamente $V_{DD}/2$ por causa do divisor de tensão formado pelas resistências de 1 kΩ.

Esta tensão está na faixa indeterminada para a maioria das séries CMOS (veja Tabela 8-10), e, portanto, é inaceitável para acionar outros dispositivos. Além disso, a corrente que flui através dos dois MOSFETs condutores será muito maior do que o normal, especialmente para valores mais altos de V_{DD}, e pode danificar os CIs.

É relativamente fácil reconhecer quando duas saídas CMOS foram colocadas em curto, porque os sinais terão três níveis diferentes: (1) ALTO quando ambas as saídas estiverem em ALTO, (2) BAIXO quando as duas estiverem em BAIXO, e (3) aproximadamente $V_{DD}/2$ quando as saídas estiverem em níveis diferentes. Isto está ilustrado na Fig. 8-42.

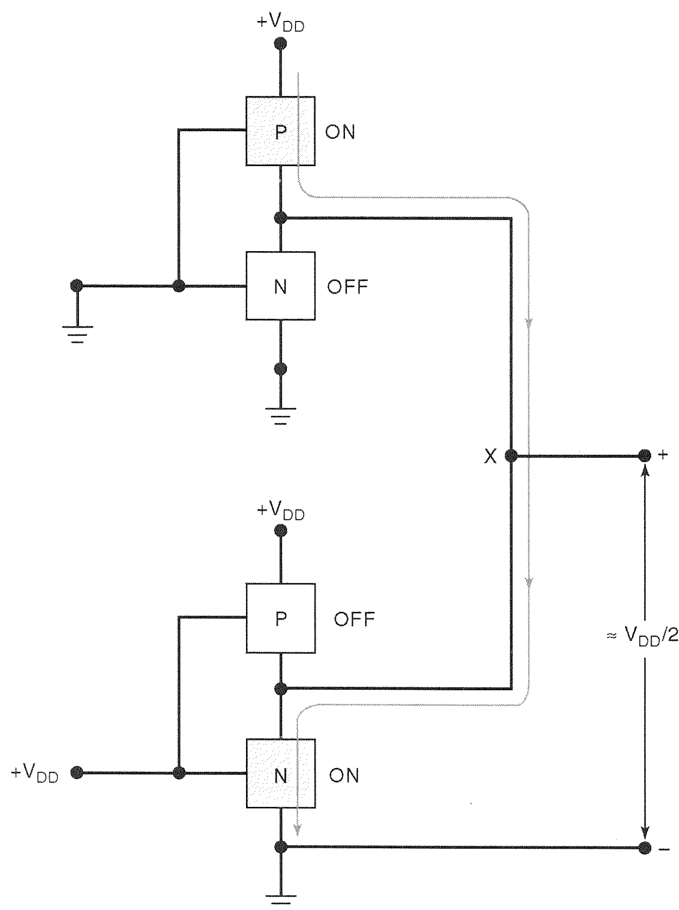


Fig. 8-41 - Quando saídas CMOS são colocadas em curto, o valor da tensão no terminal de saída comum será de aproximadamente $V_{DD}/2$ se as saídas estiverem em níveis diferentes.

Saídas em Dreno Aberto

Alguns dispositivos CMOS estão disponíveis com saídas em dreno aberto, que são correspondentes às saídas em coletor aberto dos dispositivos TTL. Nestes dispositivos, o estágio de saída consiste apenas em um MOSFET canal-N cujo dre-

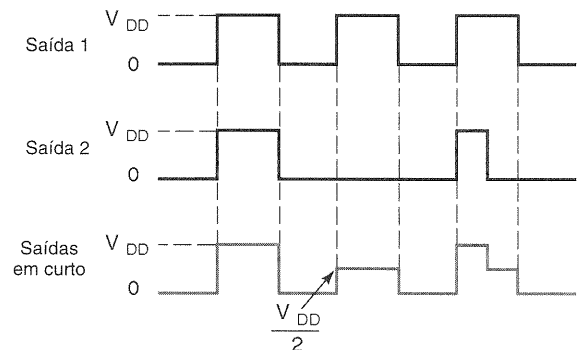


Fig. 8-42 - Se duas entradas CMOS comuns são colocadas em curto, elas estarão com um valor de tensão aproximadamente igual a $V_{DD}/2$ quando estiverem em níveis diferentes.

no não está conectado, uma vez que o MOSFET canal-P foi eliminado. Um resistor de pull-up é necessário para produzir o nível de tensão do estado ALTO. Do mesmo modo que as saídas em coletor aberto, as saídas em dreno aberto podem implementar a conexão wired-AND. A Fig. 8-43(a) mostra três INVERSORes com saídas em dreno aberto 74HC05 conectados em um arranjo wired-AND.

Saídas Tristate

Diversos CIs CMOS têm saídas tristate, cuja operação é similar à das saídas tristate TTL. Tudo que foi dito sobre saídas tristate TTL pode ser aplicado às CMOS. As saídas de terceiro estado CMOS podem ser conectadas em um barramento, desde que apenas uma única saída esteja habilitada em cada instante. A Fig. 8-43(b) mostra três buffers tristate 74HC125 conectados em um arranjo de barramento.

Questões de Revisão

1. Qual é a indicação de que duas saídas convencionais CMOS estão em curto?
2. O que é uma saída em dreno aberto?

8-18 PORTA DE TRANSMISSÃO CMOS (CHAVE BILATERAL)

Um circuito especial CMOS que não tem nenhum equivalente em TTL ou em ECL é a **porta de transmissão** ou **chave bilateral**, que age essencialmente como uma chave de um pólo e uma posição controlada por um nível lógico de entrada. Esta porta de transmissão permite que os sinais passem em ambas as direções e é bastante útil em aplicações digitais e analógicas.

A Fig. 8-44(a) mostra o arranjo básico de uma chave bilateral. Ela consiste em um MOSFET-P e em um MOSFET-N em paralelo, de modo que qualquer polaridade de tensão de entrada possa ser comutada. A entrada de CONTROLE e seu inverso são usados para abrir (OFF) e fechar (ON) a chave. Quando o CONTROLE está ALTO, ambos os MOSFETs estão ligados e a chave está fechada. Quando o CONTROLE é BAIXO, ambos os MOSFETs estão desligados e a chave está aberta. Idealmente, este circuito opera como um relé eletromecânico. Na prática, entretanto, ele não é um curto-circuito perfeito quando a chave está fechada, pois a resistência R_{ON} da chave é geralmente igual a 200 Ω. No estado aberto, a resistência da chave é muito alta, em geral 10^{12} Ω, o que para a maioria das aplicações é um circuito aberto. O símbolo na Fig. 8-44(b) é usado para representar a chave bilateral.

Este circuito é chamado chave *bilateral* porque os terminais de entrada e saída são intercambiáveis. Os sinais aplicados à entrada da chave podem ser digitais ou analógicos, desde que permaneçam entre os limites de 0 e V_{DD} volts.

A Fig. 8-45(a) mostra o diagrama lógico tradicional para o CI 4016 com quatro chaves bilaterais, que também está disponível na série 74HC como 74HC4016. As quatro chaves bilaterais deste CI operam conforme foi descrito anteriormente. Cada chave é controlada independentemente por sua própria entrada de controle. Por exemplo, o estado CONDUZINDO/NÃO CONDUZINDO (ON/OFF) da chave superior é controlado pela entrada $CONT_A$. Uma vez que as chaves são bidirecionais, qualquer um dos terminais da chave pode ser usado como entrada ou saída, como podemos ver pelas indicações.

O símbolo IEEE/ANSI para este CI pode ser visto na Fig. 8-45(b). Cada pequeno retângulo contém uma chave bilateral, mas, como de costume, a notação de dependência IEEE é colocada apenas no triângulo superior. A entrada de controle recebe a denominação X1. O símbolo "X" é usado para denominar uma entrada que controla a transmissão

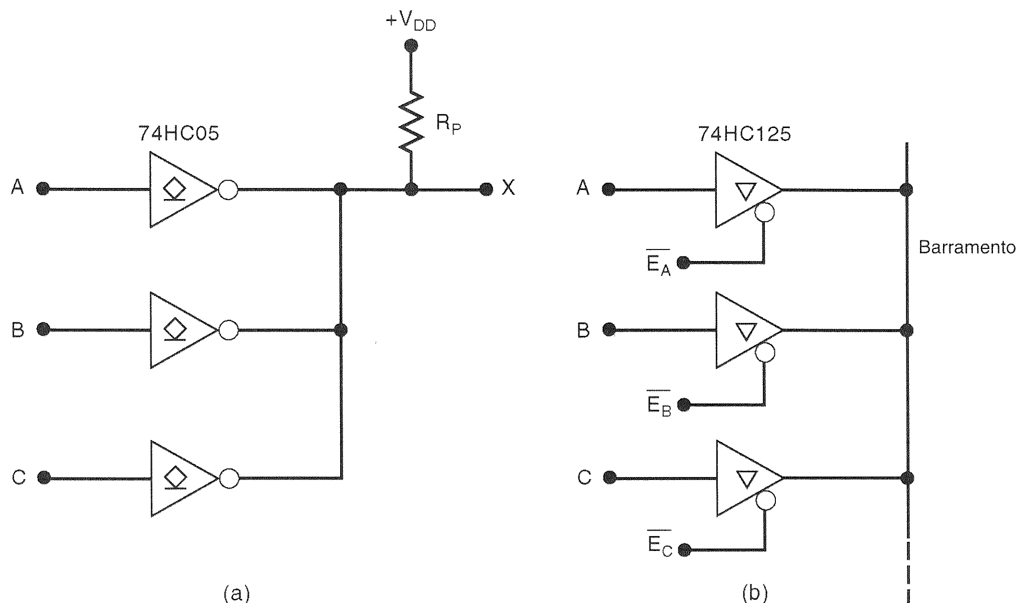


Fig. 8-43 - (a) Componentes dreno aberto CMOS em conexão wired-AND; (b) saídas CMOS tristate em um arranjo de barramento.

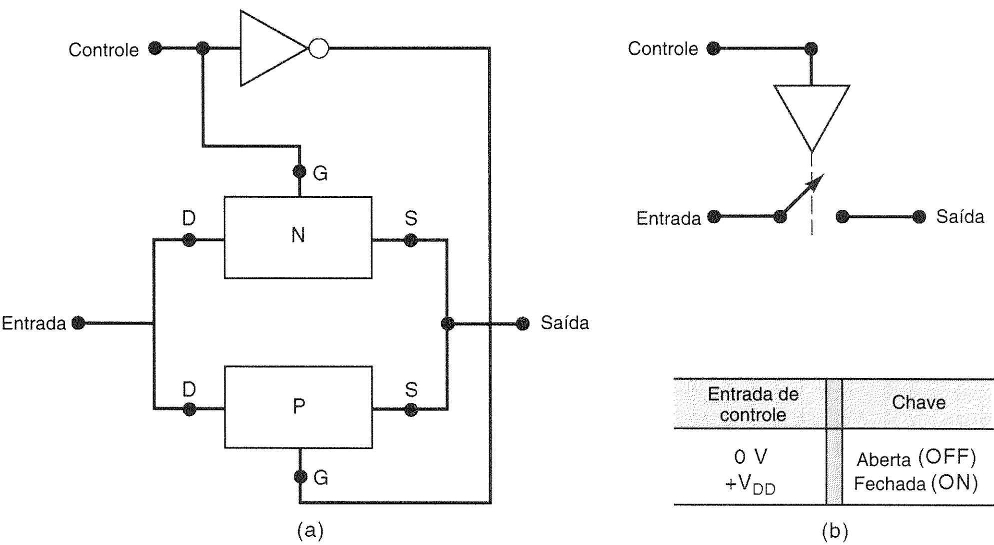


Fig. 8-44 - Chave bilateral CMOS (porta de transmissão).

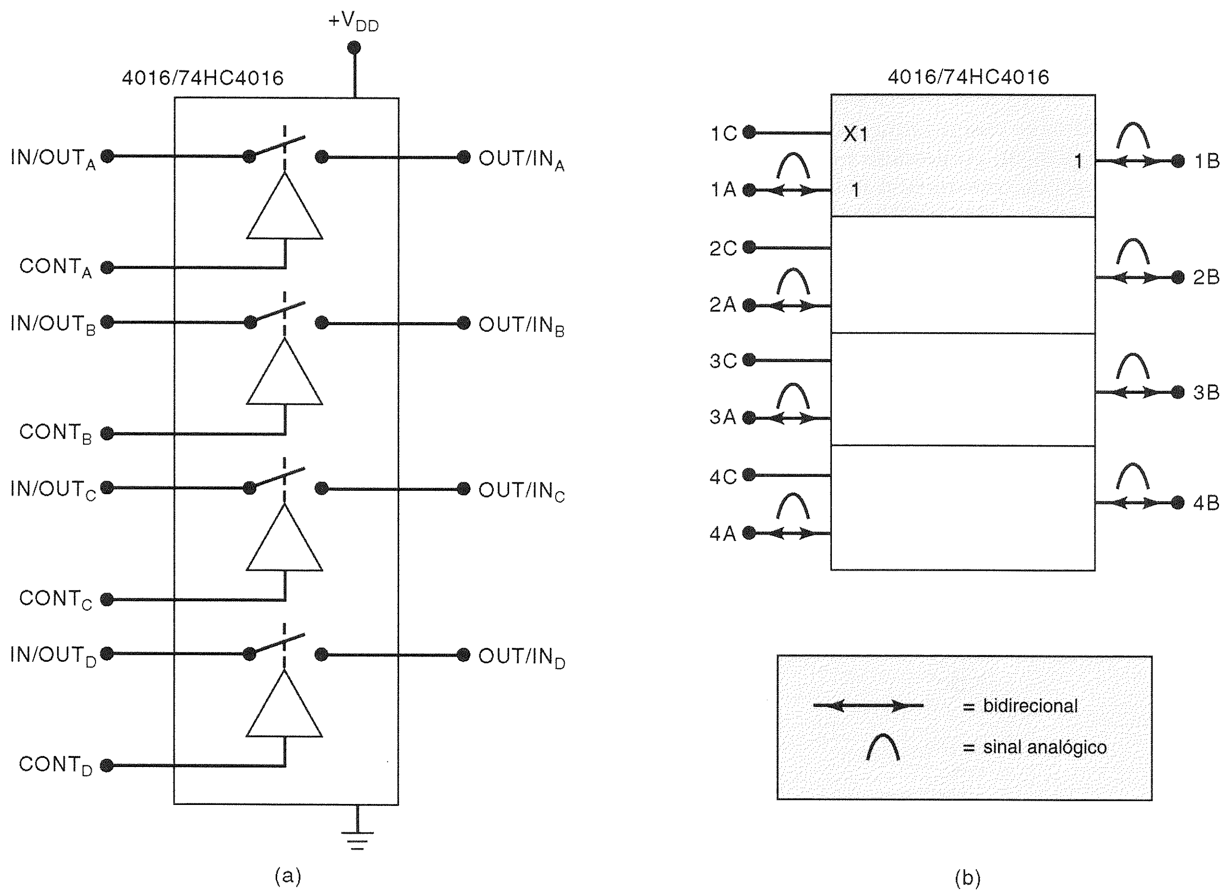


Fig. 8-45 - O CI 4016/74HC4016, que contém 4 chaves bilaterais: (a) símbolo lógico tradicional; (b) símbolo IEEE/ANSI.

bidirecional entre terminais de entrada/saída que têm a mesma numeração. O "1" na denominação X1 indica que esta entrada controla a transmissão entre os terminais com denominação interna 1. Do lado de fora do retângulo, uma seta bidirecional pode ser vista nos terminais da chave para indicar sua operação bidirecional. A "ferradura" é usada para indicar que estes terminais podem lidar com sinais analógicos e não estão restritos a níveis digitais.

EXEMPLO 8-12

Descreva a operação do circuito da Fig. 8-46.

Solução

Neste exemplo, duas das chaves bilaterais estão conectadas de modo que uma entrada analógica comum possa ser comutada para *X* ou para *Y*, dependendo do estado lógico da entrada SELEÇÃO. Quando esta entrada está em BAIXO, a chave superior está fechada e a de baixo está aberta, de modo que V_{IN} está conectada à saída *X*. Quando SELEÇÃO está em ALTO, a chave superior está aberta e a inferior está fechada, de modo que V_{IN} está conectada à saída *Y*. A Fig. 8-46(b) mostra algumas das formas de onda típicas. Observe que para funcionar corretamente V_{IN} deve estar dentro do intervalo de 0 V a + V_{DD}.

A chave bilateral 4016/74HC4016 pode comutar tensões de entrada que estejam dentro do intervalo que vai de 0 V a + V_{DD} e, portanto, não pode ser usada para sinais que são tanto positivos quanto negativos em relação a terra. Os CIs 4316 e 74HC4316 são chaves bilaterais quádruplas que podem comutar sinais *bipolares*. Estes componentes têm uma outra fonte de alimentação chamada V_{EE}, que pode ser negativa em relação a terra. Isto permite que os sinais de entrada possam estar no intervalo entre V_{DD} e V_{EE}. Por exemplo, com V_{EE} = − 5 V e V_{DD} = + 5V, o sinal de entrada analógico pode ter qualquer valor entre − 5 V a + 5 V.

Questões de Revisão

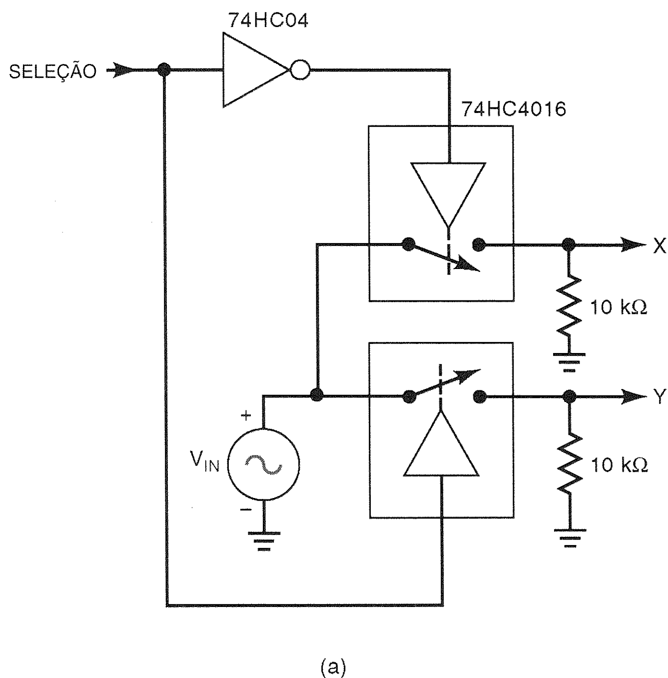
1. Descreva a operação de uma chave bilateral CMOS.
2. *Verdadeiro ou falso*: Não existe chave bilateral TTL.

8-19 INTERFACEAMENTO DE CIs

Interfaceamento significa conectar a(s) saída(s) de um circuito ou sistemas na(s) entrada(s) de um outro circuito que tem características elétricas diferentes. Geralmente, não pode ser feita uma conexão direta porque existem diferenças nas características elétricas entre o circuito *acionador*, que está fornecendo o sinal de saída, e o circuito de *carga*, que está recebendo o sinal.

Um *circuito de interface* é aquele que está conectado entre o acionador e a carga. Sua função é receber o sinal de saída do acionador e condicioná-lo de modo a torná-lo compatível com os requisitos da carga.

Nas próximas seções, estudaremos os problemas envolvendo o interfaceamento de dispositivos de uma família lógica com uma outra diferente. Este tipo de interface geralmente ocorre em sistemas digitais mais complexos, em que os projetistas utilizam diferentes famílias lógicas para diferentes partes do sistema para poderem tirar vantagens dos pontos fortes de cada família. Por exemplo, as TTL de alta velocidade (74AS, 74S) poderiam ser usadas nas partes do sistema que devem operar em freqüências mais altas, a



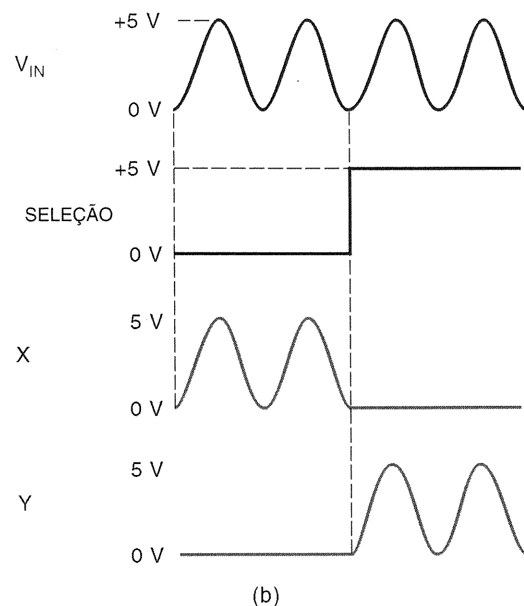


Fig. 8-46 - Exemplo 8-12: duas chaves bilaterais 74HC4016 usadas para direcionar um sinal analógico para duas saídas diferentes.

TABELA 8-13 Valores de corrente de entrada/saída para componentes padrão com tensão de alimentação de 5 V.

	CMOS				TTL				
Parâmetro	**4000B**	**74HC/HCT**	**74AC/ACT**	**74AHC/AHCT**	**74**	**74LS**	**74AS**	**74ALS**	**74F**
I_{IH} (max)	1 μA	1 μA	1 μA	1 μA	40 μA	20 μA	200 μA	20 μA	20 μA
I_{IL} (max)	1 μA	1 μA	1 μA	1 μA	1,6 mA	0,4 mA	2 mA	100 mA	0,6 mA
I_{OH} (max)	0,4 mA	4 mA	24 mA	8 mA	0,4 mA	0,4 mA	2 mA	400 mA	1,0 mA
I_{OL} (max)	0,4 mA	4 mA	24 mA	8 mA	16 mA	8 mA	20 mA	8 mA	20 mA

74HCT nas partes mais lentas do sistema, e N-MOS para as partes LSI e VLSI.

CIs da mesma família lógica são projetados para serem conectados juntos sem que qualquer consideração especial seja feita, desde que a limitação de fan-out de cada saída não seja ultrapassada. Quando você conecta a saída de um CI à entrada de outro CI de uma família lógica diferente, ou de uma série diferente em uma mesma família, você deve se preocupar com os parâmetros de tensão e corrente dos dois dispositivos. Isto, geralmente, faz com que você verifique nos manuais dos componentes os parâmetros de corrente e tensão de entrada e saída. A Tabela 8-10, que temos usado, tem os parâmetros de tensão de entrada e saída de CIs de várias séries. Estes valores serão, de um modo geral, válidos para a maioria dos componentes nas séries indicadas. A Tabela 8-13 mostra os valores de corrente de entrada/saída para componentes *padrão* de diversas séries CMOS ou TTL, isto é, para CIs que não possuam nenhum circuito especial de entrada e saída. Estes valores não são válidos para componentes como buffers, que possuem uma capacidade de corrente de saída maior, ou para CIs cuja entrada externa está internamente conectada a mais de uma porta no chip. Usaremos estas tabelas nas seções a seguir para demonstrar as idéias e procedimentos de interfaceamento de CIs; entretanto, deve ficar entendido que, na prática, é melhor consultar as folhas de características de cada componente para obter os valores corretos de tensão e corrente.

8-20 TTL ACIONANDO CMOS

Quando interfaceamos tipos diferentes de CIs, devemos verificar se o dispositivo acionador pode satisfazer os requisitos de tensão e corrente do dispositivo de carga. Examinando a Tabela 8-13, vemos que os requisitos de corrente de entrada para CMOS são extremamente baixos quando comparados com a capacidade de corrente de saída de qualquer série TTL. Portanto, dispositivos TTL não têm nenhum problema para satisfazer os requisitos de correntes de entrada dos CMOS.

Existe um problema, entretanto, quando comparamos as tensões de saída TTL com os requisitos de tensão de entrada CMOS. A Tabela 8-10 mostra que V_{OH}(min) de todas as séries TTL é muito baixa quando comparada com V_{IH}(min) das séries 4000B, 74HC e 74AC. Para estas situações, algo deve ser feito para aumentar a tensão de saída TTL para um valor que seja aceitável para um dispositivo CMOS.

A solução mais comum utilizada para este problema de interface pode ser vista na Fig. 8-47, onde a saída TTL está conectada a +5 V por um resistor de pull-up. A presença do resistor faz com que o valor da tensão de saída suba para aproximadamente 5 V no estado ALTO, e deste modo forneça o nível de entrada CMOS necessário. Este pull-up não é necessário se o dispositivo CMOS for 74HCT ou 74ACT porque estas séries foram projetadas para aceitar saídas TTL diretamente, como pode ser visto na Tabela 8-10.

TTL Acionando CMOS com Tensão de Alimentação Alta

Se o CI CMOS estiver operando com V_{DD} maior do que 5 V, a situação torna-se ainda mais difícil. Por exemplo, com V_{DD} = 10 V, a entrada CMOS necessita de um V_{IH}(min) = 7 V. As saídas de muitos componentes TTL não podem operar com uma tensão de alimentação maior do que 5 V, e portanto não é possível conectar um resistor de pull-up para 10 V. A série LS-TTL de alguns fabricantes (por exemplo, a Fairchild) pode operar com um resistor de pull-up para 10 V. De um modo geral, a folha de características deve ser verificada antes de usar um pull-up para mais do que 5 V.

Quando um resistor de pull-up não puder ser usado, existem algumas alternativas. Uma solução comum está mostrada na Fig. 8-48, onde o buffer coletor aberto 7407 é usado como interface entre a saída totem-pole TTL e o CMOS com V_{DD} > 5 V. O 7407 é buffer não-inversor similar ao 7406 e cuja tensão de saída pode chegar a 30 V.

Uma outra solução é utilizar um circuito **conversor de níveis de tensão** como o chip 40104. Este chip CMOS foi projetado para receber uma entrada de baixa tensão (por exemplo, de uma saída TTL) e convertê-la em uma tensão mais alta para a entrada CMOS.

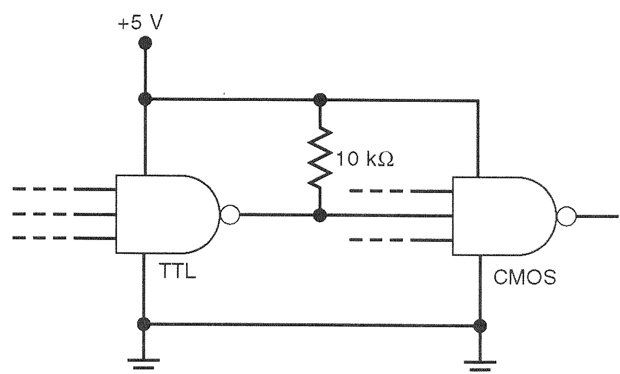


Fig. 8-47 - Um resistor de pull-up externo é usado quando um componente TTL aciona CMOS.

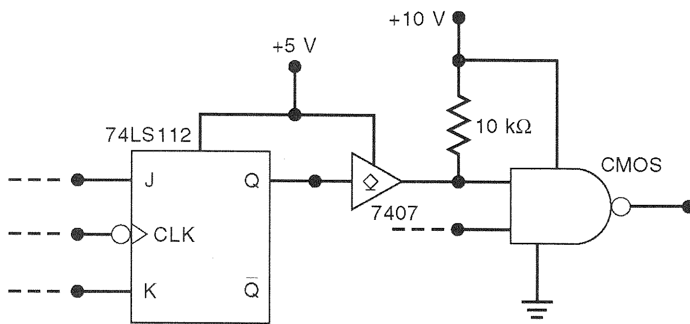


Fig. 8-48 - Um buffer coletor aberto pode ser usado para interfacear CMOS com circuitos CMOS que tenham fonte de alimentação com tensões mais altas.

Questões de Revisão

1. O que deve ser feito para interfacear uma saída TTL com uma entrada 74AC ou 74HC? Considere $V_{DD} = +5$ V.
2. Quais são as diversas maneiras de interfacear um dispositivo TTL com um outro CMOS com tensão de alimentação mais alta?

8-21 CMOS ACIONANDO TTL

Antes de estudarmos o problema de interfacear saídas CMOS com entradas TTL, vamos rever as características da saída CMOS para os dois estados lógicos.

A Fig. 8-49(a) mostra o circuito equivalente de uma saída no estado ALTO. O R_{ON} do MOSFET-P conecta o terminal de saída a V_{DD} (lembre-se de que o MOSFET-N está desligado). Então, o circuito de saída CMOS age como uma fonte de V_{DD} com uma resistência de saída igual a R_{ON}. O valor de R_{ON} se encontra, geralmente, no intervalo entre 100 e 1.000 Ω.

A Fig. 8-49(b) mostra o circuito equivalente de saída para o estado BAIXO. A R_{ON} do MOSFET-N conecta o terminal de saída à terra (lembre-se de que o MOSFET-P está desligado). Portanto, a saída CMOS age como uma pequena resistência para a terra, isto é, como um sorvedor de corrente.

CMOS Acionando TTL no Estado ALTO

A Tabela 8-10 mostra que as saídas CMOS podem facilmente fornecer tensão suficiente (V_{OH}) para satisfazer os requisitos de uma entrada TTL no estado ALTO (V_{IH}). A Tabela 8-13 mostra que saídas CMOS podem fornecer corrente mais do que suficiente para satisfazer os requisitos de corrente de entrada (I_{IH}). Logo, nenhum cuidado especial é necessário para o estado ALTO.

CMOS Acionando TTL no Estado BAIXO

A Tabela 8-13 mostra que entradas TTL têm uma corrente de entrada relativamente alta no estado BAIXO, que pode ir desde 100 μA até 2 mA. As séries 74HC e 74HCT podem sorver até 4 mA, e portanto elas não teriam problema para acionar uma *única* carga TTL de qualquer série. A série 4000B, entretanto, é bem mais limitada. Seu baixo I_{OL} não é suficiente para acionar nem mesmo uma única entrada das séries 74 ou 74AS. A série 74AHC tem capacidade de acionamento comparável à da série 74LS.

EXEMPLO 8-13

Quantas entradas 74LS podem ser acionadas por uma saída 74HC? E para uma saída 4000B?

Solução

A série 74LS tem I_{IL}(max) = 0,4 mA. O 74HC pode sorver até I_{OL} = 4 mA. Portanto, uma saída 74HC pode acionar *dez* cargas 74LS (4 mA/0,4 mA = 10).

Uma saída 4000B pode sorver apenas 0,4 mA, e, portanto, pode acionar apenas *uma* entrada 74LS.

EXEMPLO 8-14

Quantas entradas 74ALS podem ser acionadas por uma saída 74HC? E quantas entradas 74AS?

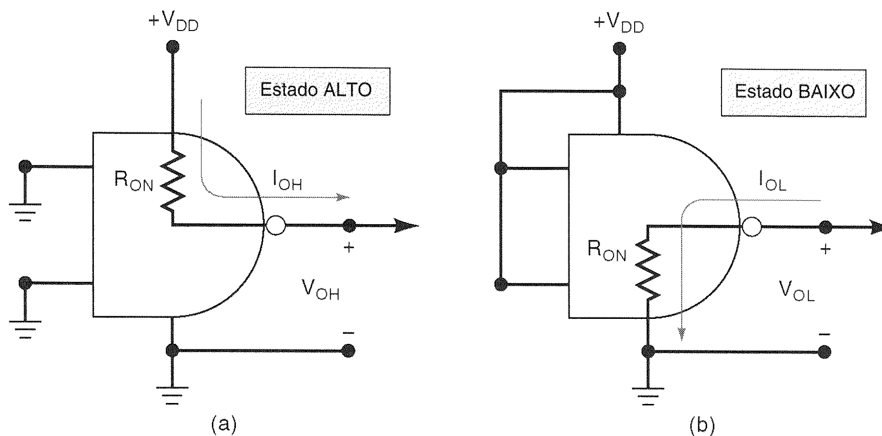


Fig. 8-49 - Circuitos equivalentes de uma saída CMOS para ambos os estados.

Solução

As entradas 74ALS têm I_{IL}(max) = 100 μA. Portanto, uma saída 74HC pode acionar *quarenta* entradas 74ALS (4 mA/100 μA = 40). Uma saída 74HC pode acionar apenas *duas* entradas 74AS (4 mA/2 mA = 2).

EXEMPLO 8-15

O que está errado no circuito da Fig. 8-50(a)?

Solução

A saída do 74HC00 pode sorver 4 mA, mas existem três entradas 74ALS que necessitam de 3 × 2 mA = 6 mA.

EXEMPLO 8-16

O que está errado no circuito da Fig. 8-50(b)?

Solução

A saída da porta NOR4001B pode sorver 0,4 mA, mas as três entradas 74LS necessitam de 3 × 0,4 mA = 1,2 mA.

Situações como aquelas mostradas na Fig. 8-50 podem ocorrer quando um equipamento mais antigo estiver sendo interfaceado com um equipamento mais novo, ou quando diversos módulos digitais estiverem sendo combinados para implementar um sistema. A solução mais simples para o caso da Fig. 8-50(a) é substituir a porta 74HC00 por uma 74AHC00. Se o componente estiver soldado na placa ou se um dispositivo de capacidade de acionamento mais alta não estiver disponível em um encapsulamento compatível pino a pino, como no caso da Fig. 8-50(b), então é necessário algum tipo de circuito de interface. Este circuito de interface deve ter uma corrente de entrada baixa e uma capacidade de fornecer corrente suficientemente alta para acionar as cargas. A Fig. 8-51 mostra uma possível interface para o caso da Fig. 8-50(b). Na Fig. 8-51, o 74LS125, que é um buffer não-inversor com saídas tristate, está permanentemente habilitado, e o 4001B consegue acioná-lo. Sua saída pode facilmente acionar cargas 74LS com a certeza de que o nível lógico 0 vai colocar as entradas LS no estado BAIXO.

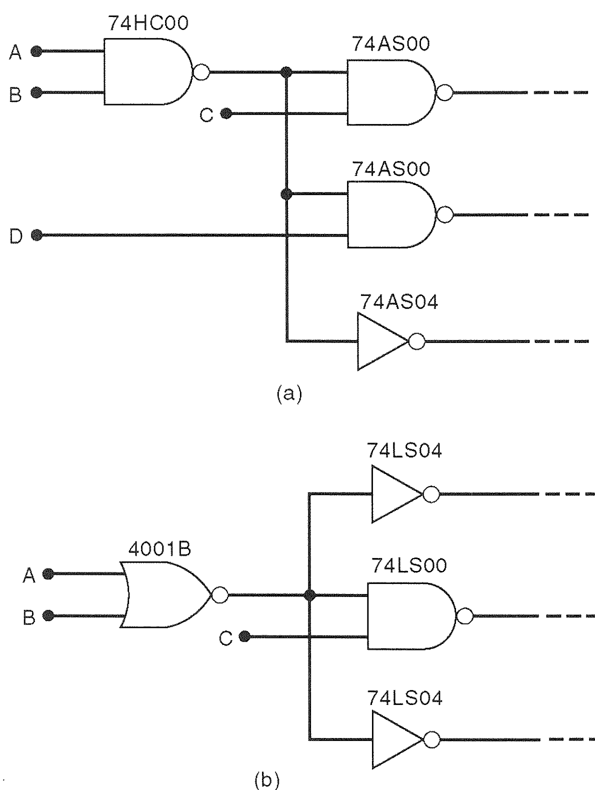


Fig. 8-50 - Exemplos 8-15 e 8-16.

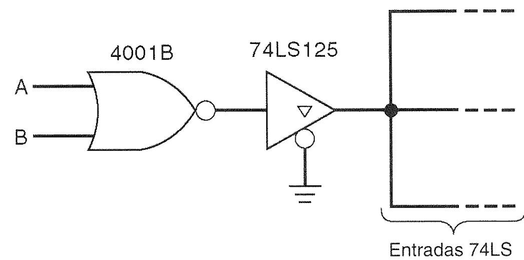


Fig. 8-51 - Um buffer é usado para interfacear componentes CMOS de baixa capacidade de corrente com entradas 74LS.

CMOS com Tensão de Alimentação Alta Acionando TTL

Alguns fabricantes de CIs têm produzido diversos componentes 74LS TTL que podem suportar tensões de entrada de até 15 V. Estes dispositivos podem ser acionados diretamente por saídas CMOS que estejam operando com V_{DD} = 15 V. A maioria das entradas TTL não suporta mais do que 7 V em suas entradas, e portanto um circuito de interface é necessário para o caso de elas serem acionadas por saídas CMOS operando com tensão de alimentação mais alta. A interface funciona como um conversor de níveis que converte uma entrada de alta tensão para uma saída de 5 V que pode ser conectada a uma entrada TTL. A Fig. 8-52 mostra como o 4050B realiza esta conversão entre 15 e 5 V.

Questões de Revisão

1. Qual é a função de um circuito de *interface*?
2. *Verdadeiro ou falso:* Todas as saídas CMOS podem acionar entradas TTL no estado ALTO.
3. *Verdadeiro ou falso:* Qualquer saída CMOS pode acionar pelo menos uma entrada TTL.
4. Que séries CMOS podem acionar TTL sem o resistor de pull-up?
5. Quantas entradas 7400 podem ser acionadas por uma saída 74HCT00?
6. Como um CMOS com tensão de alimentação mais alta pode ser interfaceado com TTL?

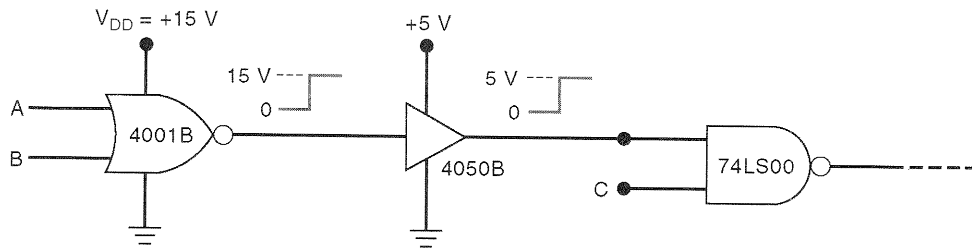


Fig. 8-52 - Um buffer 4050 também pode ser usado como um conversor de nível entre um componente TTL e um outro CMOS com fonte de alimentação de valor mais alto.

8-22 COMPARADORES DE TENSÃO

Um outro componente bastante útil para fazer a interface com circuitos digitais é o **comparador de tensão**. Ele é especialmente útil em sistemas que contêm tanto tensões analógicas quanto componentes digitais. Um comparador de tensão compara duas tensões. Se a tensão na entrada (+) for maior que a tensão na entrada (−), a saída estará em ALTO. Se a tensão na entrada (−) for maior que a tensão na entrada (+), a saída estará em ALTO. As entradas de um comparador podem ser vistas como entradas analógicas, mas a saída é digital, uma vez que ela sempre estará em ALTO ou BAIXO. Por esta razão, o comparador é muitas vezes chamado de conversor analógico-digital (A/D) de um bit. Estudaremos conversores A/D em detalhe no Cap. 10.

Um LM339 é um CI analógico que contém quatro comparadores de tensão. A saída de cada comparador é um transistor com coletor aberto, do mesmo modo que uma saída TTL em coletor aberto. O V_{CC} pode variar de 2 até 36 V, mas geralmente o valor escolhido é um pouco maior que as tensões analógicas de entrada que estão sendo comparadas. Um resistor de pull-up deve ser colocado da saída para a tensão de alimentação que os circuitos digitais usam (normalmente 5 V).

EXEMPLO 8-17

Suponha que uma incubadora deve ter um alarme de emergência para avisar quando a temperatura ultrapassa um nível perigoso. O dispositivo que mede a temperatura pode ser um LM34 que fornece uma tensão diretamente proporcional à temperatura. A tensão de saída varia 10 mV por grau Fahrenheit. O sistema digital de alarme deve soar quando a temperatura exceder 100°F. Projete um circuito que faça a interface entre o sensor de temperatura e o circuito digital.

Solução

Precisamos comparar a tensão de saída de um sensor com uma tensão limiar fixa. Em primeiro lugar, devemos calcular a tensão limiar apropriada. Queremos que a saída do comparador vá para ALTO quando a temperatura exceder 100°F. A saída do LM34 nesta temperatura será

$$100°F \cdot 10 \text{ mV/°F} = 1{,}0 \text{ V}$$

Isto significa que devemos colocar o pino de entrada (−) do comparador em 1,0 V e conectar na entrada (+) o LM34. Para criar um tensão de referência 1,0 V, vamos usar um divisor de tensão e escolher uma corrente de polarização

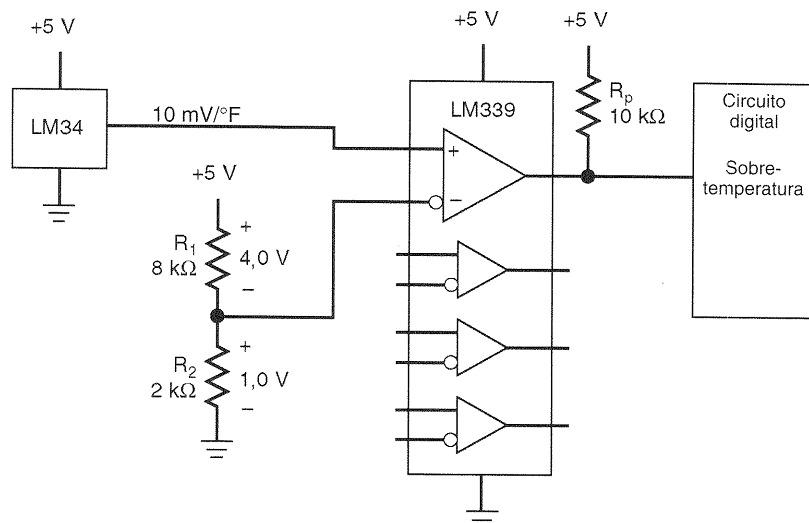


Fig. 8-53 - Circuito detector de temperatura limite usando um comparador de tensão LM339.

de 100 μA. A corrente de entrada do LM339 pode ser desprezada, uma vez que drena menos do que 1 μA. Isto significa que $R_1 + R_2$ deve ser igual a 10 kΩ. Neste exemplo, vamos usar uma fonte de +5 V. A Fig. 8-53 mostra um circuito completo. Os cálculos são mostrados a seguir:

$$V_{R2} = V_{CC} \cdot \frac{R_2}{R_1 + R_2}$$

$$R_2 = V_{R2} \cdot (R_1 + R_2)/V_{CC}$$
$$= 1{,}0\ \text{V}(10\ \text{k}\Omega)/5\ \text{V} = 2\ \text{k}\Omega$$
$$R_1 = 10\ \text{k}\Omega - R_2 = 10\ \text{k}\Omega - 2\ \text{k}\Omega = 8\ \text{k}\Omega$$

Questões de Revisão

1. O que faz a saída do comparador ir para o estado lógico ALTO?
2. O que faz a saída do comparador ir para o estado lógico BAIXO?
3. A saída do LM339 é mais parecida com uma saída TTL totem-pole ou com uma saída coletor aberto?

8-23 PESQUISA DE FALHAS

Um **pulsador lógico** é um equipamento de depuração que pode gerar pulsos de curta duração quando acionado manualmente, apenas apertando-se um botão. O pulsador mostrado na Fig. 8-54 tem sua ponta em forma de agulha, que toca o ponto do circuito onde deve ser aplicado um pulso. O pulsador é projetado para que ele perceba e determine o nível lógico presente em um ponto do circuito e produza um pulso de nível oposto. Em outras palavras, se o nível é BAIXO, o pulsador produz um pulso estreito em ALTO. Se o ponto estiver em ALTO, ele produz um pulso em nível BAIXO.

O pulsador é usado momentaneamente para mudar o nível lógico no ponto do circuito, mesmo que a saída de um outro dispositivo esteja conectada ao mesmo ponto. Na Fig. 8-54, a ponta de um pulsador está colocada no ponto *X*, que também está conectado na saída da porta NAND. O pulsador tem uma impedância de saída muito baixa (geralmente menor ou igual a 2 Ω), e, portanto, ele se sobrepõe à saída da porta NAND e pode mudar a tensão neste ponto. O pulsador, entretanto, não pode produzir um pulso de tensão em um ponto que esteja diretamente em curto com a terra ou V_{CC} (por exemplo, através de uma ponte de solda).

Usando um Pulsador e uma Ponta de Prova para Testar um Circuito

Um pulsador pode ser utilizado para injetar manualmente um pulso, ou uma série de pulsos, em um circuito para testar a resposta do mesmo. Uma ponta de prova lógica quase sempre é usada para monitorar a resposta do circuito a um pulsador. Na Fig. 8-54, a operação de comutação de um flip-flop J-K está sendo testada pela aplicação de pulsos por um pulsador e pela monitoração da saída Q com a ponta de prova lógica. A combinação pulsador/ponta de prova é muito útil para verificar a operação de um dispositivo lógico enquanto ele está conectado em um circuito. Note que o pulsador aplica um pulso ao ponto do circuito *sem* desconectar a saída da porta NAND que está acionando aquele ponto.

Descobrindo Pontos do Circuito em Curto

O pulsador e a ponta de prova podem ser usados para verificar se pontos do circuito estão diretamente em curto com a terra ou V_{CC}. Quando você coloca o pulsador e a ponta de prova em um mesmo ponto e pressiona o botão do pulsador, a ponta de prova deve indicar a ocorrência de um pulso. Se a ponta de prova indicar um nível constante BAIXO, este ponto está em curto com a terra. Se a ponta de prova indicar um nível ALTO, o ponto está conectado a V_{CC}.

Rastreador de Corrente

Um rastreador de corrente é uma ferramenta de depuração que pode detectar a *mudança* no valor da corrente em um fio ou trilha de uma placa de circuito impresso sem interromper o circuito. O rastreador de corrente tem uma ponta isolada que contém um enrolamento captador magnético. Quando esta ponta é colocada em um ponto do circuito, ela percebe a mudança no campo magnético produzida por uma mudança de corrente e faz com que um pequeno LED indicador pisque. O rastreador de corrente não responde a níveis estáticos de corrente, não importando o quão grande eles possam ser. Ele responde apenas a mudanças no nível de corrente.

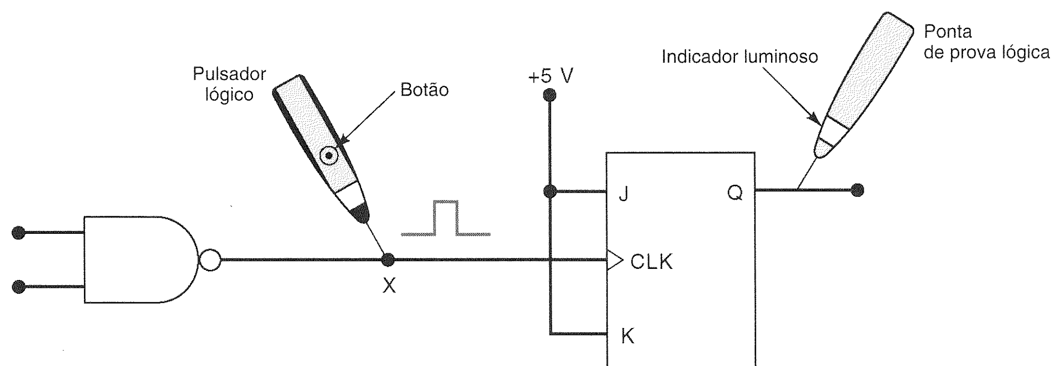


Fig. 8-54 - Um pulsador lógico pode injetar um pulso em qualquer ponto que não esteja ligado diretamente a terra ou V_{CC}.

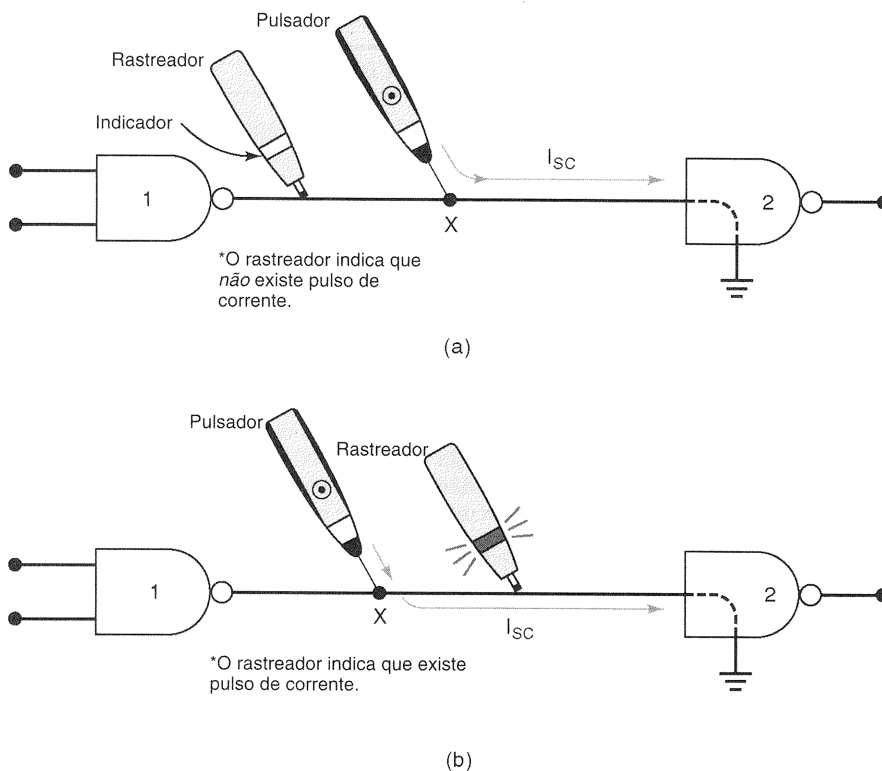


Fig. 8-55 - Um pulsador lógico e um rastreador de corrente podem ser usados para rastrear pontos em curto.

Um rastreador de corrente é usado com um pulsador para determinar a localização exata de curtos para terra ou V_{CC}. Isto está ilustrado na Fig. 8-55, onde o ponto *X* está em curto com a terra, através de um curto interno na entrada da porta 2. Se o pulsador estiver no ponto *X* e seu botão for pressionado, nenhum pulso de tensão será detectado devido ao curto para a terra. Entretanto, haverá um pulso de corrente fluindo da saída do pulsador para a terra através do curto-circuito (I_{SC}). Este pulso de corrente pode ser detectado pelo rastreador.

Na Fig. 8-55 (a), o rastreador é colocado à esquerda do ponto *X*, e um pulso é aplicado. O rastreador não indicará nenhum pulso de corrente no caminho que ele está monitorando. Na Fig. 8-55(b), o rastreador é colocado à direita de *X*. Agora, quando um pulso é aplicado, o rastreador indicará a ocorrência de um pulso no caminho que ele está monitorando. Isto prova que o curto para a terra está no interior da porta 2 em vez de estar na saída da porta 1.

A combinação rastreador/pulsador é bastante útil na depuração de circuitos onde muitas saídas coletor aberto, dreno aberto ou tristate estão conectadas em um ponto comum, que está permanentemente em BAIXO ou ALTO. Qualquer uma das saídas pode produzir uma falha, e a técnica do rastreador/pulsador usada anteriormente pode ser empregada para encontrar a saída com problema.

Questões de Revisão

1. Qual é a função de um pulsador?
2. *Verdadeiro ou falso*: Um pulsador produzirá um pulso de tensão em qualquer ponto.
3. Qual é a função de um rastreador de corrente?
4. *Verdadeiro ou falso*: Um rastreador de corrente detecta apenas variações no valor da corrente.

RESUMO

1. Todos os dispositivos lógicos digitais têm natureza semelhante, mas são bastante diferentes no que se refere aos detalhes de suas características. Uma compreensão dos termos que são usados para descrever estas características é importante e nos permite comparar a performance destes dispositivos. Compreendendo as capacidades e limitações de cada tipo de dispositivo, podemos combiná-los de modo inteligente, aproveitando os pontos positivos de cada um para construir sistemas digitais confiáveis.
2. A família lógica TTL tem sido usada nos últimos 30 anos. Seus circuitos internos utilizam transistores bipolares. Esta família oferece várias portas SSI, e dispositivos MSI. Diversas séries, de numeração semelhante, têm sido desenvolvidas à medida que os avanços na tecnologia permitem melhorias nas características.
3. Quando dispositivos devem ser conectados, é importante saber quantas entradas uma dada saída pode acionar, sem comprometer a confiabilidade. Isto é chamado de *fan-out*.
4. Saídas com coletor aberto podem ser ligadas juntas para implementar a operação wired-AND. As saídas tristate podem ser conectadas juntas para permitir que vários componentes possam dividir linhas de dados comuns em um arranjo chamado *barramento*. Neste caso, apenas um dispositivo pode colocar um nível lógico no barramento (isto é, acionar o barramento) de cada vez.
5. Os componentes lógicos *mais rápidos* são aqueles da família que usa lógica com acoplamento pelo emissor (ECL). Esta tecnologia também usa transistores bipolares, mas não é tão amplamente usada quanto a TTL devido às suas características de entrada/saída.
6. Os transistores MOSFET podem ser usados para implementar funções lógicas. As maiores vantagens da lógica MOS são o baixo consumo e a maior densidade de integração.
7. O uso de MOSFETs complementares deu origem às famílias lógicas CMOS. Elas têm se apoderado do mercado devido

ao seu baixo consumo e à velocidade de operação competitiva.

8. A contínua necessidade de reduzir consumo e tamanho tem levado os fabricantes a desenvolver uma nova linha de dispositivos que opera com 3,3 V em vez de 5 V.
9. Os dispositivos lógicos que usam tecnologias diferentes nem sempre podem ser conectados diretamente e ainda operar de modo confiável. As características de tensão e corrente das entradas e saídas devem ser consideradas para assegurar a operação correta.
10. A tecnologia CMOS permite que um sistema digital controle chaves analógicas chamadas *portas de transmissão*. Estes componentes podem permitir ou não a passagem de um sinal analógico, dependendo do nível digital que o controla.
11. Comparadores de tensão oferecem uma outra ponte entre os sinais analógicos e sistemas digitais. Estes dispositivos comparam tensões analógicas e fornecem como saída um sinal digital cujo nível depende de qual entrada é maior.

TERMOS IMPORTANTES

fan-out
produto velocidade-potência
imunidade ao ruído
margem de ruído
fornecimento de corrente
absorção de corrente
DIP
espaçamento entre os terminais
montagem em superfície
TTL
totem-pole
transistor de pull-down
transistor de pull-up
entradas em flutuação
desacoplamento da fonte de alimentação
saída coletor aberto
wired-AND
buffer/driver
tristate
contenção de barramento
lógica com acoplamento pelo emissor (*emitter-coupled logic* — ECL)
MOSFETs
P-MOS
N-MOS
CMOS
descarga eletrostática (ESD)
latch-up
porta de transmissão
chave bilateral
interfaceamento
conversor de nível de tensão
comparador de tensão
pulsador lógico

PROBLEMAS

(As folhas de características dos CIs utilizados nestes problemas podem ser encontradas no Apêndice.)

SEÇÕES 8-1 A 8-3

8-1. Dois circuitos lógicos diferentes têm as características mostradas na Tabela 8-14.

TABELA 8-14

	Circuito *A*	**Circuito *B***
V_{fonte} (V)	6	5
V_{IH} (min) (V)	1,6	1,8
V_{IL} (max) (V)	0,9	0,7
V_{OH} (min) (V)	2,2	2,5
V_{OL} (max) (V)	0,4	0,3
t_{PLH} (ns)	10	18
t_{PHL} (ns)	8	14
P_D (mW)	16	10

(a) Qual o circuito que tem a melhor imunidade a ruído em estado BAIXO? E qual o que tem a melhor imunidade a ruído em estado ALTO?
(b) Qual circuito pode operar em freqüências mais altas?
(c) Qual circuito consome mais corrente?

8-2. Veja no Apêndice as folhas de características dos CIs e use os valores *máximos* para determinar a potência dissipada média P_D(med), t_{pd} (med) e o produto velocidade-potência para uma porta de cada um dos seguintes CIs TTL. (Veja Exemplo 8-2 na Seção 8-3.)
(a) 7432
(b) 74S32
(c) 74LS20
(d) 74ALS20
(e) 74AS20

8-3. Um acerta família lógica tem os seguintes parâmetros de tensão:

$V_{IH}(\text{min}) = 3{,}5$ V $V_{IL}(\text{max}) = 1{,}0$ V
$V_{OH}(\text{min}) = 4{,}9$ V $V_{OL}(\text{max}) = 0{,}1$ V

(a) Qual é o maior pulso espúrio positivo que ela pode tolerar?
(b) Qual é o maior pulso espúrio negativo que ela pode tolerar?

QUESTÃO DE FIXAÇÃO

8-4. Para cada afirmação, indique o termo ou parâmetro que está sendo descrito.
(a) A corrente em uma entrada quando o nível lógico 1 é aplicado.
(b) A corrente drenada de V_{CC} quando todas as saídas estão em BAIXO.
(c) Intervalo de tempo necessário para uma saída comutar de 1 para 0.
(d) Uma medida bastante comum usada para comparar a performance de diferentes famílias lógicas de CIs.
(e) O tamanho do spike de tensão que pode ser tolerado em uma entrada em nível ALTO sem causar operação indeterminada.
(f) Um encapsulamento para CI que não exige furos na placa de circuito impresso.
(g) Quando uma saída em BAIXO recebe corrente de um circuito de entrada que ela está acionando.
(h) O número de entradas diferentes que uma saída pode acionar com segurança.
(i) Arranjo de transistores de saída em um circuito TTL padrão.
(j) Outro termo para descrever o transistor de pull-down Q_4.
(k) Intervalo de valores de V_{CC} permitido para a série TTL 74.
(l) V_{OH}(min) e V_{IH}(min) para a série 74.
(m) V_{IL}(max) e V_{OL}(max) para a série 74.

(n) Quando uma saída em ALTO fornece corrente para uma carga.

SEÇÃO 8-4

8-5. (a) A partir da Tabela 8-6, determine as margens de ruído quando um componente 74LS está acionando uma entrada 74ALS.

(b) Repita o item (a) para um 74ALS acionando 74LS.

(c) Qual será a margem de ruído total de um circuito lógico que usa uma combinação de circuitos 74LS e 74ALS?

(d) Um certo circuito lógico tem V_{IL}(max) = 450 mV. Que séries TTL podem ser usadas com este circuito?

SEÇÕES 8-5 E 8-6

8-6. QUESTÃO DE FIXAÇÃO

(a) Defina *fan-out*.

(b) Em que tipo de portas entradas conectadas juntas sempre contam como somente uma carga em estado BAIXO?

(c) Defina entradas em "flutuação".

(d) O que causa pulsos estreitos de corrente em TTL? Que efeito indesejável eles podem produzir? O que pode ser feito para reduzir este efeito?

(e) Quando uma saída TTL aciona uma entrada TTL, de onde vem I_{OL}? Para onde I_{OH} vai?

8-7. Veja a folha de características do flip-flop J-K 74LS112.

(a) Determine as correntes de carga das entradas J e K em estado ALTO e BAIXO.

(b) Determine as correntes de carga das entradas clock e clear em estado ALTO e BAIXO.

(c) Quantas entradas de clock de um flip-flop J-K 74LS112 podem ser acionadas por uma saída do mesmo flip-flop?

8-8. A Fig. 8-56(a) mostra um flip-flop J-K 74LS112 cuja saída deve acionar 8 entradas TTL padrão. Uma vez que isto excede o fan-out do 74LS112, algum tipo de buffer é necessário. A Fig. 8-56(b) mostra a possibilidade de usar uma porta NAND de um 74LS37 (CI NAND buffer quádruplo), que possui um fan-out muito maior do que o 74LS112. Observe que, como a saída Q está sendo usada, a porta NAND está agindo como um inversor. Veja a folha de características do 74LS37.

(a) Determine seu fan-out máximo para TTL padrão.

(b) Determine sua máxima absorção de corrente em estado BAIXO.

D

8-9. Portas tipo buffer são geralmente mais caras que portas comuns, e às vezes portas comuns não-utilizadas podem ser usadas para resolver um problema de carga como o da Fig. 8-56(a). Mostre como portas NAND 74LS00 podem ser usadas para resolver o problema.

8-10. Veja o diagrama lógico da Fig. 8-57, onde a saída de um EX-OR 74LS86 está acionando várias 74LS20. Determine se o fan-out do 74LS86 está sendo excedido, e explique. Repita o exercício usando agora os componentes da série 74ALS.

8-11. Quanto tempo demora para que a saída típica de um 74LS04 mude de estado em resposta para uma transição positiva em sua entrada?

8-12. Para o circuito da Fig. 8-57, determine qual o maior tempo possível para que uma mudança na entrada A seja sentida na saída W. Suponha o pior caso e valores máximos para os atrasos de propagação. (*Sugestão*: Lembre-se de que portas NAND são portas inversoras.) Repita o exercício usando agora apenas CIs 74ALS.

8-13. (a) A Fig. 8-58 mostra um contador 74LS193 com sua entrada de reset principal (MR) ativa em ALTO controlada por uma chave. O resistor R é usado para colocar MR em BAIXO quando a chave estiver aberta. Qual é o valor máximo que pode ser usado para R?

(b) Repita o item (a) para o 74ALS193.

C, T

8-14. A Fig. 8-59(a) mostra um circuito que é usado para converter uma senóide de 60 Hz em um sinal de 60 pps, que pode disparar de modo confiável FFs e contadores. Este tipo de circuito pode ser usado em um relógio digital.

(a) Explique a operação do circuito.

(b) Um estudante está testando este circuito e observa que a saída do 74LS14 permanece em BAIXO. Ele verifica a forma de onda na entrada do INVERSOR, e ela aparece como está mostrado na Fig. 8-59(b). Pensando que o INVERSOR está com problemas, ele substitui o chip e observa os mesmos resultados. O que você acha que está causando o problema, e como ele pode ser consertado? (*Sugestão*: Examine a forma de onda v_x cuidadosamente.)

T

8-15. Para cada uma das formas de onda da Fig. 8-60, determine *por que* elas não vão ativar, de modo confiável, a entrada *CLK* de um 74LS112.

T

8-16. Uma estudante monta um circuito para ser testado. À medida que ela testa a operação do circuito, verifica que vários FFs e contadores estão sendo disparados de modo irregular. Como toda boa estudante, ela verifica o valor de V_{CC} com um voltímetro e obtém como leitura 4,97 V, que é aceitável para TTL. Então, passa a verificar todas as ligações e a substituir os CIs um por um, mas o problema persiste. Finalmente, decide observar a linha de V_{CC} com um osciloscópio e vê a forma de onda mostrada na Fig. 8-61. Qual a causa provável do ruído em V_{CC}? O que a estudante esqueceu de colocar quando montou o circuito?

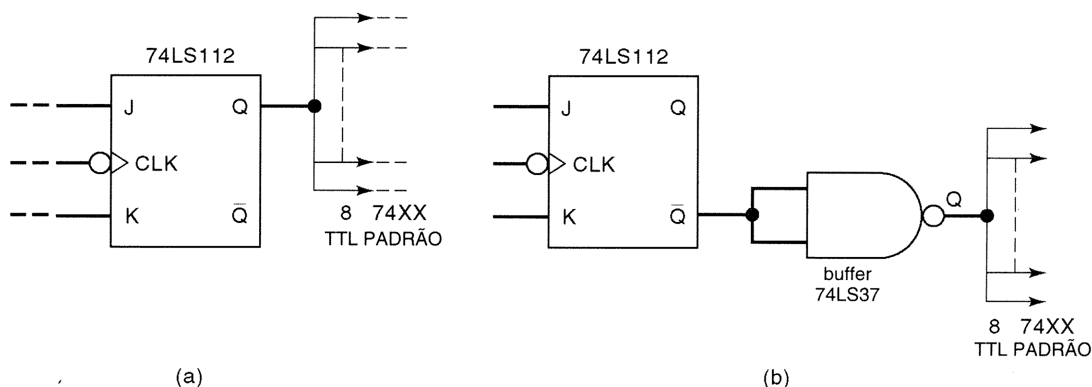


Fig. 8-56 - Problemas 8-8 e 8-9.

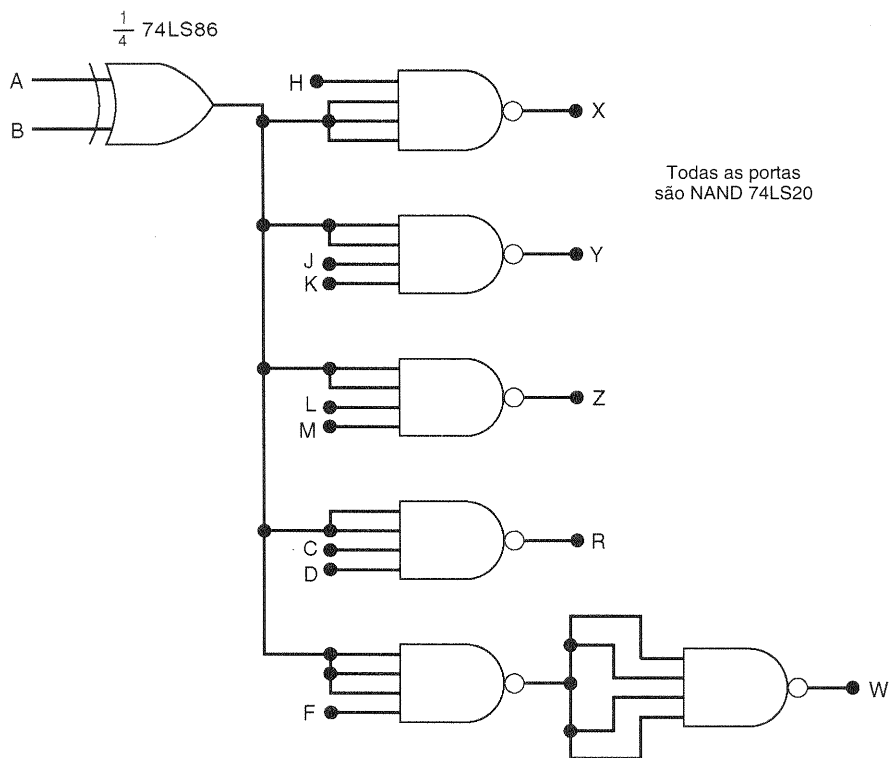


Fig. 8-57 - Problemas 8-10 e 8-12.

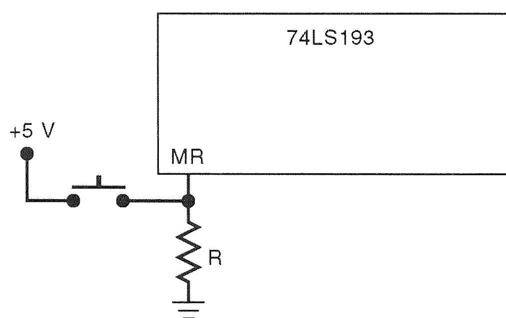


Fig. 8-58 - Problema 8-13.

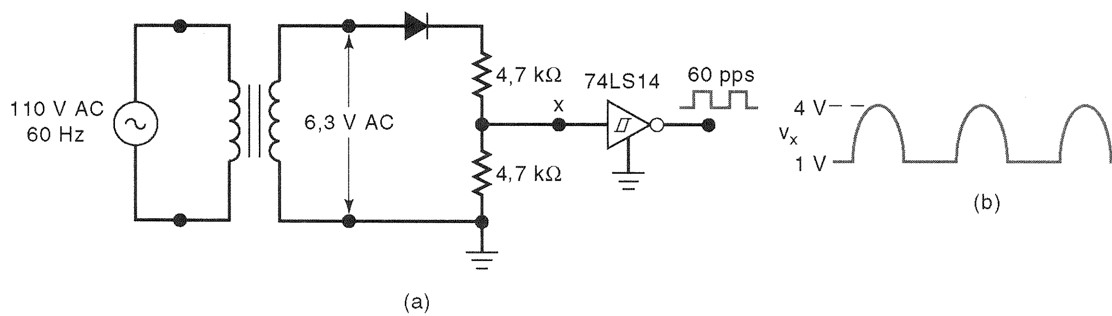


Fig. 8-59 - Problema 8-14.

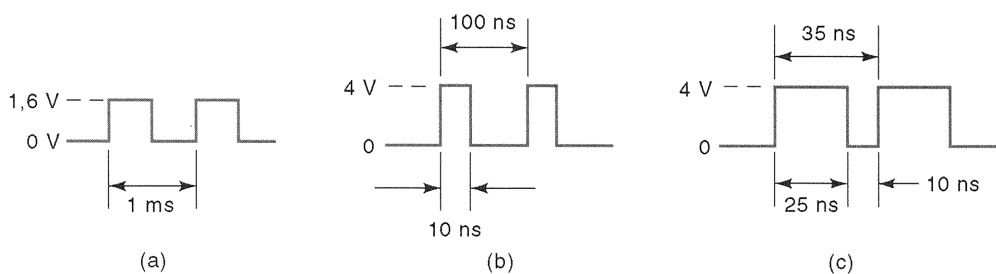


Fig. 8-60 - Problema 8-15.



Fig. 8-61 - Problema 8-16.

SEÇÕES 8-7 E 8-8

8-17. QUESTÃO DE FIXAÇÃO
- **(a)** Defina *wired-AND*.
- **(b)** O que é resistor de pull-up? Por que ele é usado?
- **(c)** Que tipos de saídas TTL podem ser ligadas juntas com segurança?
- **(d)** O que é contenção de barramento?

D

8-18. O TTL 74LS09 é um CI AND quádruplo com saídas coletor aberto. Mostre como 74LS09s podem ser usados para implementar a operação $x = A \cdot B \cdot C \cdot D \cdot E \cdot F \cdot G \cdot H \cdot I \cdot J \cdot K \cdot L \cdot M$.

8-19. Determine a expressão lógica para a saída *X* na Fig. 8-62.

8-20. Qual das situações a seguir provavelmente destruiria uma saída TTL totem-pole enquanto ela estivesse indo de ALTO para BAIXO?
- **(a)** Conectar a saída a + 5 V.
- **(b)** Conectar a saída à terra.
- **(c)** Aplicar uma entrada de 7 V.
- **(d)** Conectar outra saída TTL totem-pole.

8-21. A Fig. 8-63(a) mostra um 7406, um buffer inversor de coletor aberto, usado para controlar o estado LIGADO/DESLIGADO de um LED para indicar o estado da saída *Q* de um FF. A especificação nominal do LED é $V_F = 2{,}4$ V com $I_F = 20$ mA, e $I_F(\text{max}) = 30$ mA.
- **(a)** Que tensão aparecerá na saída do 7406 quando $Q = 0$?
- **(b)** Escolha um valor apropriado de resistor para a operação correta.

8-22. Na Fig. 8-63(b), o 7406 está sendo usado para comutar a corrente de um relé.
- **(a)** Que tensão aparecerá na saída do 7406 quando $Q = 0$?
- **(b)** Qual é a maior corrente que pode ser usada para acionar o relé?
- **(c)** Como podemos modificar este circuito para utilizar um 7407?

8-23. A Fig. 8-64 mostra como dois buffers tristate podem ser usados para construir um *buffer bidirecional* que permite que dados sejam transmitidos em ambas as direções (*A* para *B* ou *B* para *A*). Descreva a operação do circuito para os dois estados da entrada DIREÇÃO.

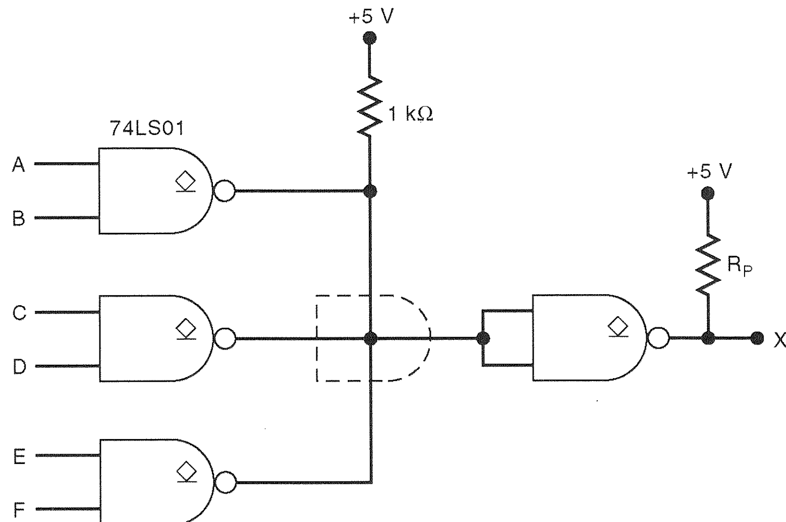


Fig. 8-62 - Problema 8-19.

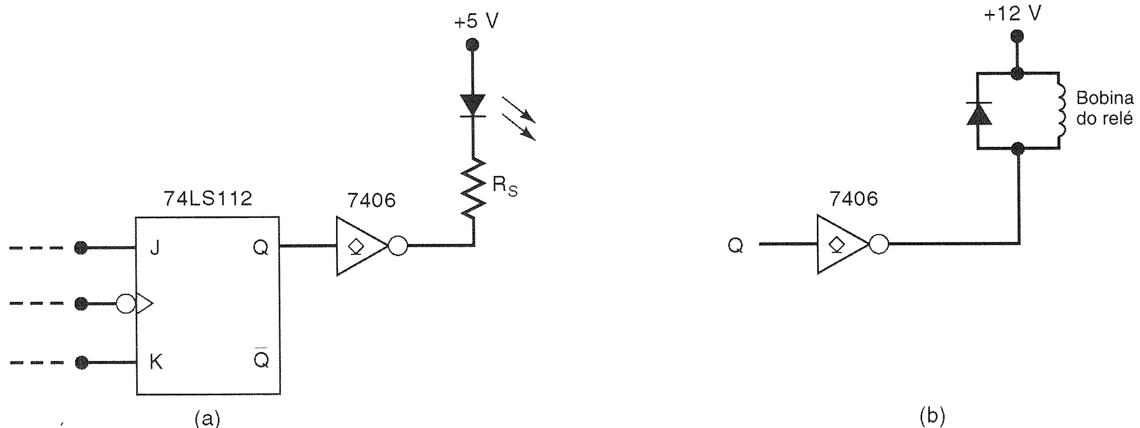


Fig. 8-63 - Problemas 8-21 e 8-22.

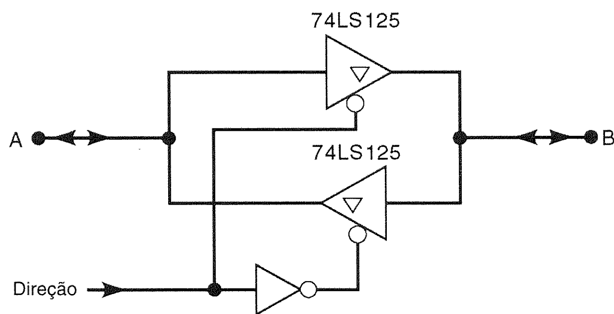


Fig. 8-64 - Problema 8-23.

8-24. O circuito da Fig. 8-65 é usado para gerar as entradas de habilitação para o circuito da Fig. 8-27.

(a) Determine que entrada de dados (A, B ou C) aparecerá no barramento para cada combinação de entradas x e y.

(b) Explique por que o circuito não vai funcionar se a porta NOR for trocada por um EX-NOR.

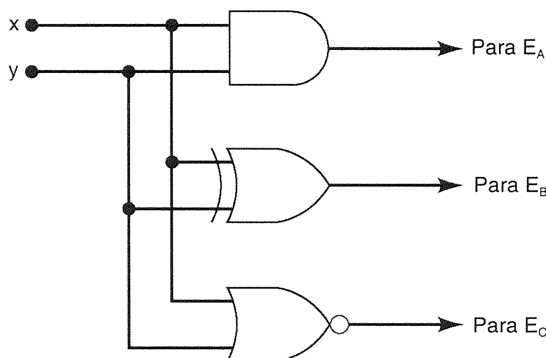


Fig. 8-65 - Problema 8-24.

SEÇÕES 8-10 A 8-13

8-25. O circuito da Fig. 8-66 é uma porta lógica N-MOS. Determine o tipo de porta que ela é. Use +16 V = nível 1 e 0 V = nível 0.

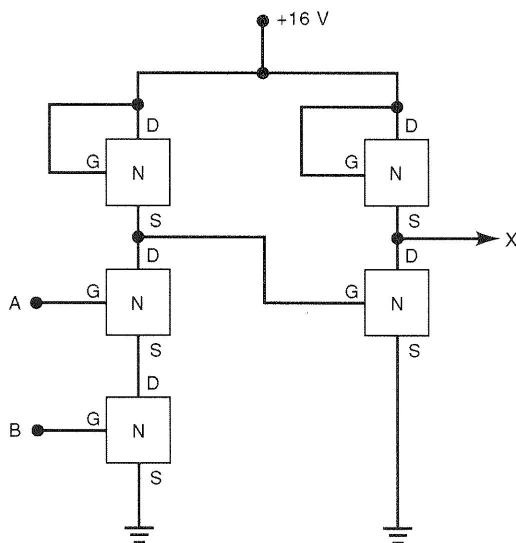


Fig. 8-66 - Problema 8-25.

8-26. Desenhe o circuito para uma porta OR N-MOS.

8-27. Quais dos itens a seguir representam vantagens da lógica N-MOS sobre a TTL?

(a) Maior densidade de integração.
(b) Maior velocidade de operação.
(c) Maior fan-out.
(d) Mais indicado para aplicações LSI.
(e) Menor consumo de potência.
(f) Saídas complementares.
(g) Maior imunidade ao ruído.
(h) Processo de fabricação mais simples.
(i) Um maior número de funções MSI e SSI.
(j) Necessidade de uma fonte de alimentação menor.

SEÇÕES 8-14 E 8-15

8-28. Quais dos itens a seguir representam vantagens que a lógica CMOS geralmente tem sobre a TTL?

(a) Maior densidade de integração.
(b) Maior velocidade.
(c) Maior fan-out.
(d) Menor impedância de saída.
(e) Processo de fabricação mais simples.
(f) Mais indicado para LSI.
(g) Menor dissipação de potência (abaixo de 1 MHz).
(h) Transistores como único elemento.
(i) Menor capacitância de entrada.
(j) Menos susceptível a ESD.

8-29. Quais das seguintes condições de operação, provavelmente, resultarão na menor dissipação média de potência em um sistema lógico CMOS? Explique.

(a) V_{DD} = 5 V, freqüência de comutação máxima f_{max} = 1 MHz
(b) V_{DD} = 5 V, f_{max} = 10 kHz
(c) V_{DD} = 10 V, f_{max} = 10 kHz

8-30. A saída de cada INVERSOR em um 74LS04 está acionando duas entradas de um 74HCT08. A entrada de cada INVERSOR está em BAIXO 99% do tempo. Qual é a potência máxima que o chip 74LS04 está dissipando?

8-31. Use os valores da Tabela 8-10 para calcular a margem de ruído em estado ALTO quando uma porta 74HC está acionando uma entrada 74LS.

8-32. O que causa o latch-up em um CI CMOS? O que pode acontecer nesta condição? Que precauções devem ser tomadas para impedir o latch-up?

8-33. Veja a folha de características do CI NAND 74HC20 no Apêndice. Use os valores máximos para calcular P_D(med), t_{pd}(med) e o produto velocidade-potência. Compare com os valores calculados no Problema 8-2 para TTL.

SEÇÕES 8-17 E 8-18

8-34. Determine os valores aproximados de V_{OUT} para ambos os estados da entrada CONTROLE.

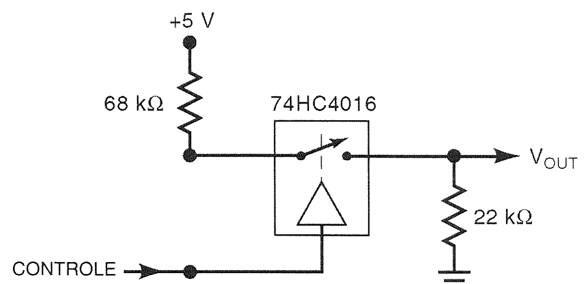
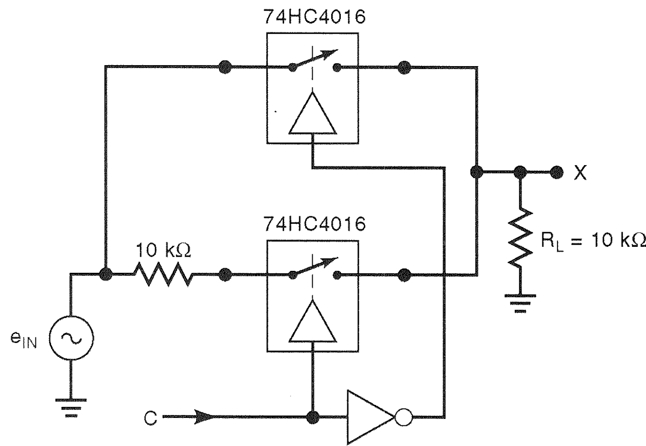


Fig. 8-67 - Problema 8-34.



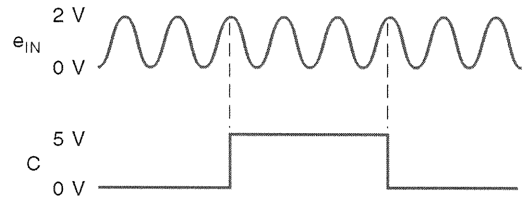


Fig. 8-68 - Problema 8-35.

8-35. Determine a forma de onda da saída *X*, na Fig. 8-68, para as formas de onda de entrada. Assuma que $R_{ON} \approx 200\ \Omega$ para a chave bilateral.

N, D, C

8-36. Determine o ganho do circuito com o amplificador operacional da Fig. 8-69 para os dois estados da entrada GANHO. Este circuito mostra o princípio básico do controle digital da amplificação de um sinal.

SEÇÕES 8-19 A 8-21

8-37. QUESTÃO DE FIXAÇÃO

(a) Quais as séries CMOS que podem ter suas entradas acionadas diretamente por uma saída TTL?

(b) Qual é a função de um conversor de nível? Quando ele é usado?

(c) Por que um buffer é necessário entre alguns tipos de saídas CMOS e entradas TTL?

(d) *Verdadeiro ou falso*: A maioria das saídas CMOS tem problema para fornecer corrente para uma entrada TTL no nível ALTO.

T

8-38. Veja a Fig. 8-70(a), onde uma saída 74LS TTL, *Q*, está acionando um INVERSOR CMOS que opera com V_{DD} = 10 V. As formas

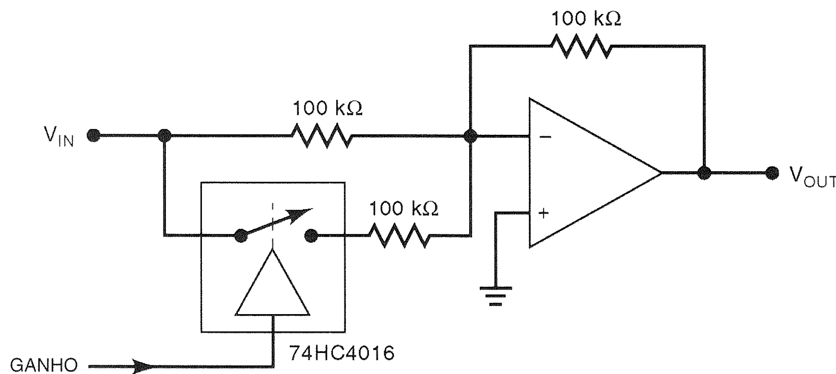


Fig. 8-69 - Problema 8-36.

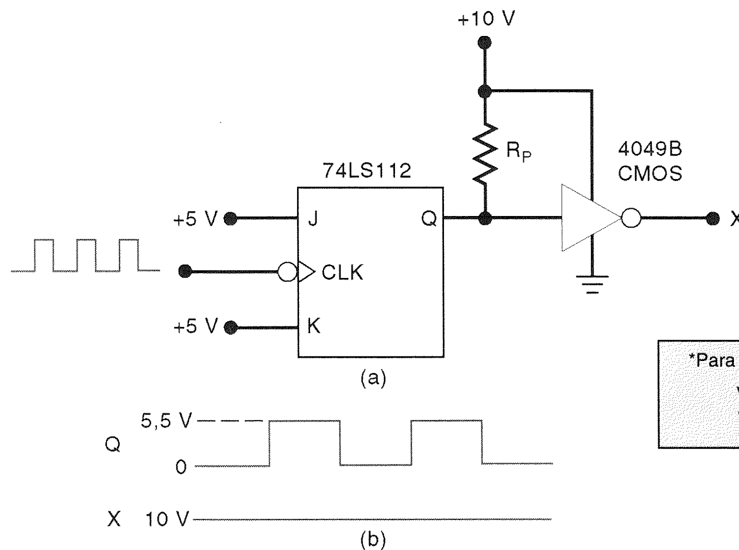


Fig. 8-70 - Problema 8-38.

de onda em Q e X são mostradas na Fig. 8-70(b). Qual das afirmações a seguir explica por que X permanece em ALTO?
(a) A alimentação de 10 V está com problema.
(b) O resistor de pull-up está muito alto.
(c) A saída do 74LS112 não tolera 10 V e mantém o nível de 5,5 V no estado ALTO. Este valor está no intervalo indeterminado para uma entrada CMOS.
(d) A entrada CMOS está carregando a saída TTL.

8-39. (a) Use a Tabela 8-13 para determinar quantas entradas 74AS podem, tipicamente, ser acionadas por uma saída 4000B.
(b) Repita o item (a) para uma saída 74HC.

T
8-40. A Fig. 8-71 é um circuito lógico que foi mal projetado. Ele contém pelo menos oito situações nas quais as características dos CIs não foram levadas em conta adequadamente. Encontre o maior número destas situações que puder.

T
8-41. Repita o Problema 8-40 com as seguintes mudanças no circuito:
- Cada CI TTL padrão foi substituído por um 74LS equivalente.
- O 4001B foi substituído por um 74HCT02.

8-42. Use a Tabela 8-13 para explicar por que o circuito da Fig. 8-72 não está funcionando como deveria. Como o problema pode ser corrigido?

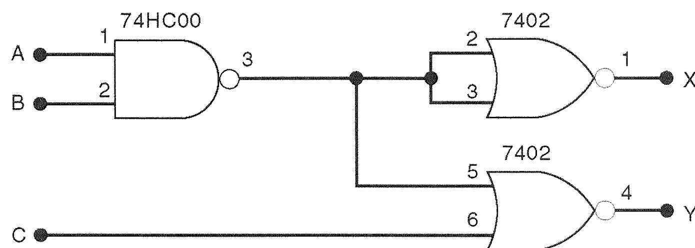


Fig. 8-72 - Problema 8-42.

SEÇÃO 8-22

D
8-43. O tanque de gasolina do seu carro tem uma unidade de transmissão do nível de combustível que funciona como um potenciômetro. Uma bóia se move para cima ou para baixo de acordo com o nível de gasolina, mudando o valor do resistor variável e produzindo uma tensão proporcional ao nível de gasolina. Um tanque cheio produz 12 V, e um tanque vazio produz 0 V. Projete um circuito usando um LM339 que ligue uma lâmpada indicadora "Nível Baixo de Combustível" quando a tensão proveniente da unidade transmissora estiver abaixo de 0,5 V.

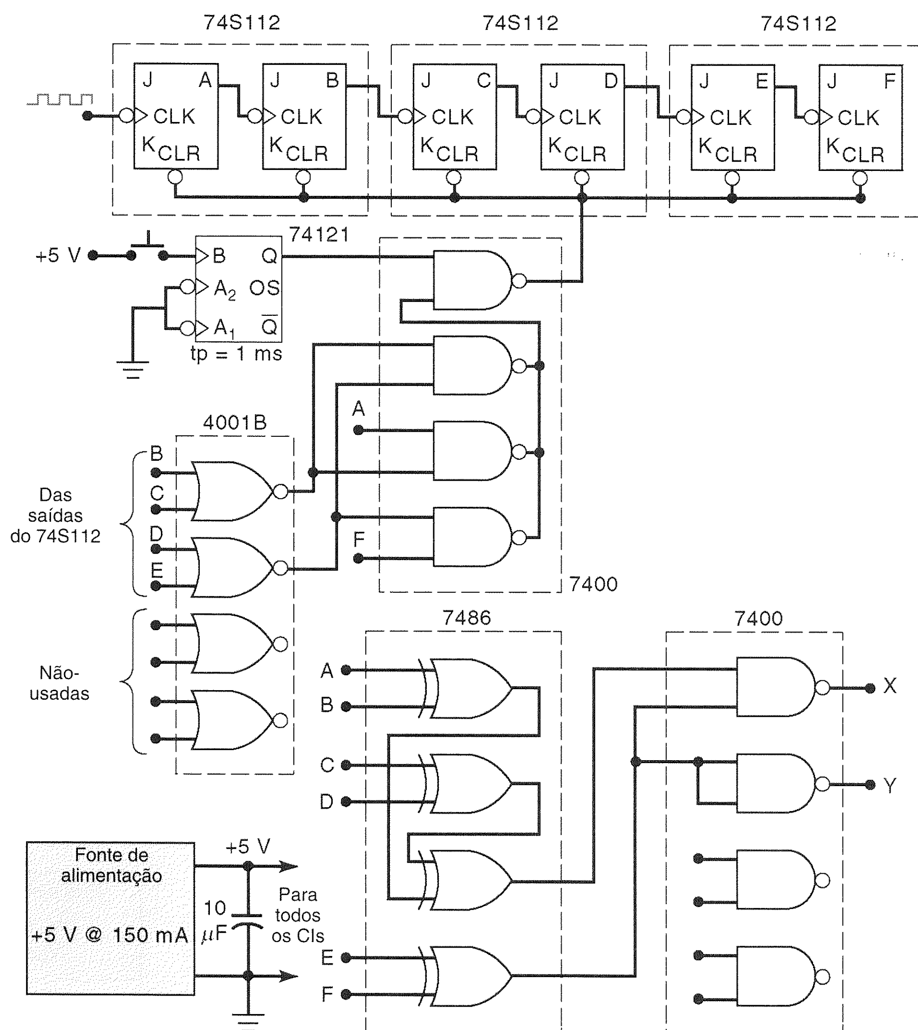


Fig. 8-71 - Problemas 8-40 e 8-41.

D

8-44. O circuito comparador de temperatura na Fig. 8-53 foi modificado substituindo-se o sensor de temperatura LM34 pelo LM35, que fornece 10 mV por grau Celsius. O alarme deve ser ativado quando a temperatura for maior do que 100°F, que é aproximadamente igual a 38°C. Calcule os novos valores de R_1 e R_2 para completar a modificação.

SEÇÃO 8-23

T

8-45. O circuito na Fig. 8-73 usa um CI 74HC05 que contém seis INVERSORes com dreno aberto. Os INVERSORes estão conectados em um arranjo wired-AND. Usando um pulsador e uma ponta de prova lógica, determinou-se que o ponto X está permanentemente em nível ALTO. Descreva um procedimento que use o rastreador de corrente para isolar a falha.

T

8-46. O circuito da Fig. 8-54 tem uma ponte de solda entre a terra e algum outro ponto entre a saída da porta NAND e a entrada do FF. Descreva um procedimento para determinar a localização da ponte de solda.

T

8-47. Na Fig. 8-47, uma ponta de prova lógica indica que a extremidade inferior do resistor de pull-up está permanentemente em estado BAIXO. Usando um pulsador lógico e um rastreador de corrente, um estudante detecta uma corrente no fio que conecta a saída TTL ao resistor. Quais das afirmações a seguir pode ser a causa da falha?

(a) R_p está aberto.

(b) O transistor que absorve a corrente na porta TTL tem um curto entre coletor e emissor.

(c) Existe uma interrupção na ligação entre R_p e a porta CMOS.

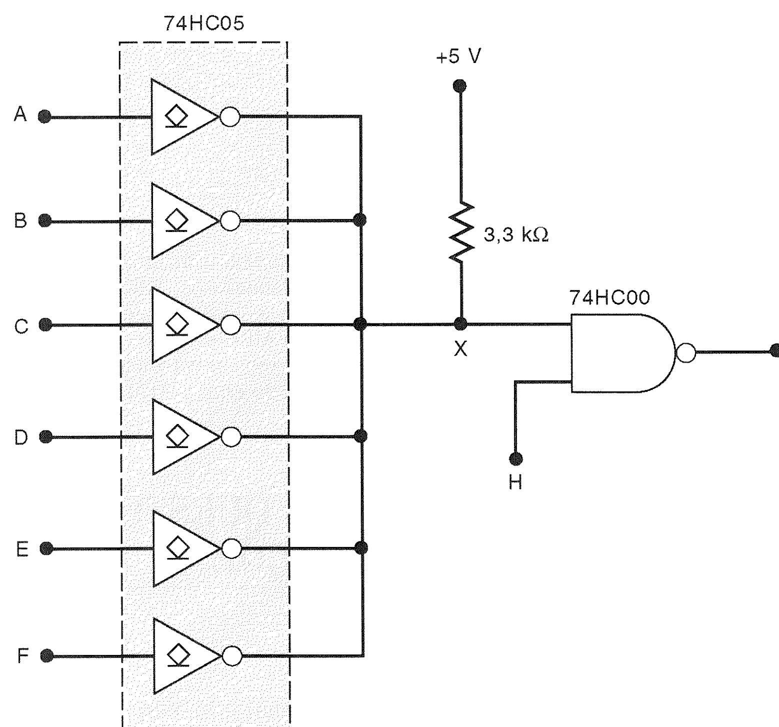


Fig. 8-73 - Problema 8-45.

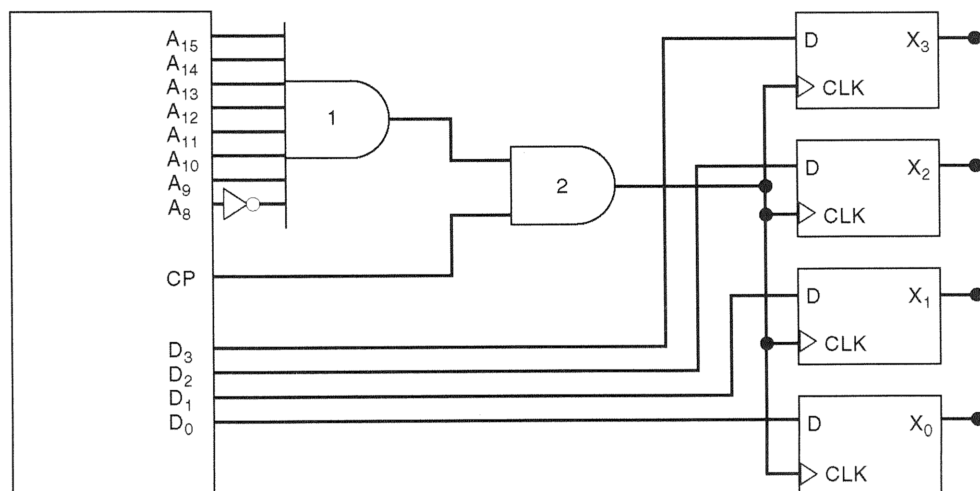


Fig. 8-74 - Problema 8-49.

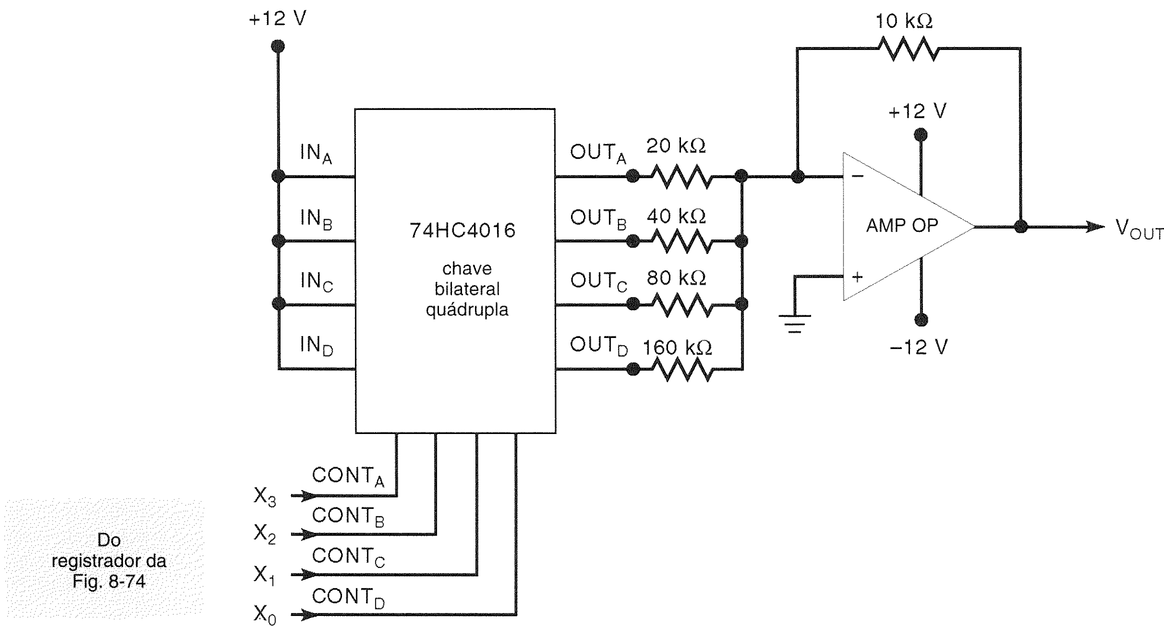


Fig. 8-75 - Problema 8-49.

T

8-48. Na Fig. 8-47, uma ponta de prova lógica indica que a extremidade inferior de R_p está permanentemente em ALTO. Um pulsador lógico e um rastreador de corrente são usados para detectar uma corrente na ligação entre R_p e a porta CMOS. Qual é a falha mais provável?

APLICAÇÃO EM MICROCOMPUTADOR

C, N

8-49. Na Seção 5-20, Fig. 5-50, vimos como um microprocessador (MPU) transfere dados para um registrador externo, sob o controle de um programa. Este diagrama é repetido na Fig. 8-74. Uma vez que os dados estejam armazenados no registrador, podem ser utilizados quando necessário. Às vezes, cada bit individual do registrador tem uma função única. Por exemplo, em computadores de automóveis, cada bit poderia representar o estado de uma variável física diferente sendo monitorada pela MPU. Um bit poderia indicar quando a temperatura do motor estivesse muito alta. Outro bit poderia indicar que a pressão do óleo está muito baixa. Em outras aplicações, os bits em um registrador são usados para produzir uma tensão analógica, que pode ser usada para acionar dispositivos que necessitam de entradas analógicas que tenham muitos níveis de tensão diferentes.

TABELA 8-15

Tempo (μs)	Dados do Microprocessador
0	0000
10	0010
20	0100
30	0111
40	1010
50	1110
60	1111
70	1111
80	1110
90	1100
100	1000

A Fig. 8-75 mostra como podemos usar uma MPU para gerar uma tensão analógica, utilizando os dados do registrador da Fig. 8-74 para controlar as entradas de um amplificador somador. Suponha que a MPU está executando um programa que está transferindo uma nova palavra para o registrador a cada 10 μs, de acordo com a Tabela 8-15. Esboce a forma de onda de V_{OUT}.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 8-1

1. Veja o texto.
2. Falso
3. Falso; V_{NH} é a diferença entre V_{OH}(min) e V_{IH}(min).
4. Falso
5. Absorção de corrente: a saída recebe (absorve) corrente da entrada do circuito que está acionando. Fornecimento de corrente: a saída fornece corrente ao circuito que está acionando.
6. DIP
7. PLCC
8. Seus pinos são dobrados
9. Não

SEÇÃO 8-2

1. Verdadeiro
2. BAIXO
3. Comutação mais rápida, menor dissipação de potência; grande spike de corrente durante a comutação de BAIXO para ALTO
4. Q_3
5. Q_6
6. Não tem transistor com múltiplos emissores.

SEÇÃO 8-4

1. **(a)** 74AS
(b) 74S, 74LS
(c) 74H, 74L

(d) 74S, 74LS, 74AS, 74ALS
(e) 74ALS

2. Todos podem operar a 40 MHz, mas o 74ALS193 terá o menor consumo.
3. Q_4, Q_5, respectivamente.

SEÇÃO 8-5

1. A resistência na condução de Q_4 e o V_{OL}(max).
2. 12
3. Suas tensões de saída podem não permanecer nos intervalos permitidos para 0 e 1 lógico.
4. Dois; cinco

SEÇÃO 8-6

1. Em nível BAIXO
2. Conectá-la a $+V_{CC}$ por um resistor de 1 kΩ; conectá-la a uma outra entrada usada.
3. Conectá-la a terra; conectá-la a uma outra entrada usada.
4. Falso; apenas no estado BAIXO.
5. Conectando pequenos capacitores de RF entre V_{CC} e terra próximos ao CI TTL, para filtrar os spikes de tensão causados pelas variações rápidas de corrente durante as transições de BAIXO para ALTO.

SEÇÃO 8-7

1. Quando saídas TTL são conectadas juntas.
2. Porque uma corrente que danifica os transistores pode fluir e porque V_{OL} excede V_{OL}(max).
3. O coletor do transistor que absorve a corrente, Q_4, não está conectado (não existe Q_3).
4. Para produzir o nível V_{OH}.
5. $\overline{A}\ \overline{B}\ \overline{C}\ \overline{D}\ \overline{E}\ \overline{F}$
6. Não tem transistor de pull-up ativo.
7. Veja a Fig. 8-24.

SEÇÃO 8-8

1. ALTO, BAIXO e Alta impedância
2. Alta impedância
3. Quando duas ou mais saídas tristate estão conectadas a um barramento comum e habilitadas ao mesmo tempo.
4. $E_A = E_B = 0$, $E_C = 1$
5. Veja a Fig. 8-28.

SEÇÃO 8-9

1. (a) Verdadeiro
(b) Verdadeiro
(c) Falso
(d) Verdadeiro
(e) Falso
(f) Falso

SEÇÃO 8-13

1. Vantagens: baixo consumo; maior densidade; processo de fabricação mais simples e barato. Desvantagens: mais lento; mais sensível a descarga elestrostática.
2. Capacitâncias de entrada das portas causam um aumento na capacitância de carga para cada entrada adicional. Esta capacitância de carga total aumenta os tempos de comutação.
3. Os altos valores de R_{OUT} e C_{LOAD}.
4. Alta densidade de integração devido à estrutura simples do circuito.
5. Veja o texto.
6. Veja o texto.

SEÇÃO 8-14

1. CMOS usa MOSFETs tanto canal-P quanto canal-N.
2. Um
3. Seis

SEÇÃO 8-15

1. 74C, HC, HCT, AHC, AHCT
2. 74ACT, HCT, AHCT
3. 74C, HC/HCT, AC/ACT, AHC/AHCT
4. BiCMOS
5. Máximo atraso de propagação permitido; capacitância de entrada de cada carga.
6. Veja a Seção 8-13.
7. CMOS
8. (a) Verdadeiro
(b) Falso
(c) Falso
(d) Falso
(e) Verdadeiro
(f) Falso

SEÇÃO 8-16

1. Um maior número de circuitos em um chip; maior velocidade de operação.
2. Não pode operar com tensões altas: um aumento na dissipação de potência pode superaquecer o chip.
3. O mesmo que para TTL padrão: 2,0 V.
4. 74ALVC, 74LV
5. 74LVT

SEÇÃO 8-17

1. A tensão da saída comum estará na faixa indeterminada.
2. O estágio de saída consiste apenas em um MOSFET canal-N com o dreno desconectado.

SEÇÃO 8-18

1. O nível lógico nas entradas de controle determinam o estado aberto/fechado da chave bidirecional que pode permitir a passagem de sinais analógicos em qualquer direção.
2. Verdadeiro

SEÇÃO 8-20

1. Um resistor de pull-up conectado entre V_{CC} e a saída TTL.
2. Um buffer de alta tensão (7407) ou um conversor de nível (40104) deveria ser conectado entre a saída TTL e a entrada CMOS.

SEÇÃO 8-21

1. Ele recebe o nível lógico de saída do circuito acionador e o condiciona de modo a torná-lo compatível com os requisitos da carga.
2. Verdadeiro
3. Falso; por exemplo, a série 4000B não pode absorver I_{IL} de um componente 74 ou de um 74AS.
4. 74HCT e ACT
5. Dois
6. Através de um conversor de nível como o 4050B.

SEÇÃO 8-22

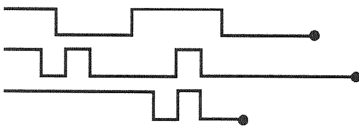
1. $V^{(+)} > V^{(-)}$
2. $V^{(-)} > V^{(+)}$
3. Coletor aberto

SEÇÃO 8-23

1. Ele injeta um pulso com determinado nível em um ponto que não esteja em curto com V_{CC} ou terra.
2. Falso
3. Ele indica a ocorrência de um pulso de corrente.
4. Verdadeiro

CAPÍTULO 9

Circuitos Lógicos MSI



SUMÁRIO

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Analisar e usar decodificadores e codificadores em diversos tipos de aplicações de circuitos.
- Comparar as vantagens e desvantagens de LEDs e LCDs (*Liquid Crystal Displays*).
- Utilizar a técnica de observação/análise para pesquisar falhas em circuitos digitais.
- Compreender a operação de multiplexadores e demultiplexadores analisando várias aplicações de circuitos.
- Comparar dois números binários utilizando um circuito comparador de magnitude.
- Entender a função e a operação dos conversores de código.
- Relacionar as precauções que devem ser consideradas para conectar circuitos digitais usando o conceito de barramento de dados.
- Interpretar a notação usada nos símbolos IEEE/ANSI para vários dispositivos MSI.

■ INTRODUÇÃO

Os sistemas digitais obtêm dados codificados em binário e informações que são, de algum modo, continuamente submetidas a operações. Algumas destas operações são: (1) *decodificação e codificação;* (2) *multiplexação;* (3) *demultiplexação;* (4) *comparação;* (5) *conversão de código;* (6) *implementação de barramentos de dados*. Todas estas operações, e outras, têm sido facilitadas pela disponibilidade de numerosos CIs na categoria MSI (*medium-scale-integration*).

Neste capítulo estudamos muitos tipos comuns de dispositivos MSI. Para cada tipo, iniciamos uma breve discussão dos princípios básicos de operação, e então apresentamos CIs específicos. Logo após, mostramos como eles podem ser usados, em separado ou em conjunto, com outros CIs em diversas aplicações.

9-1 DECODIFICADORES

Um **decodificador** é um circuito lógico que aceita um conjunto de entradas que representa um número binário e ativa somente uma saída que corresponde ao número da entrada. Em outras palavras, um circuito decodificador analisa as suas entradas, determina qual número binário está presente e ativa a saída correspondente a esse número; todas as outras saídas permanecem desativadas. O diagrama para um decodificador geral com N entradas e M saídas é mostrado na Fig. 9-1. Como cada uma das N entradas pode ser 0 ou 1, existem 2^N combinações ou códigos de entrada

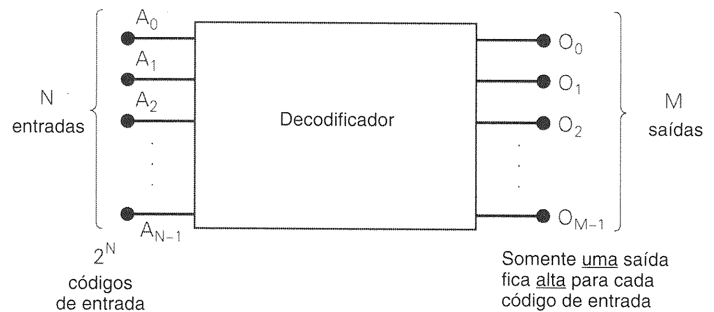


Fig. 9-1 Diagrama de um decodificador genérico.

possíveis. Para cada uma destas combinações de entrada, apenas uma das M saídas estará ativa (ALTO); todas as outras saídas estarão em BAIXO. Muitos decodificadores são projetados para produzir saídas ativas em BAIXO, onde apenas a saída selecionada fica em BAIXO e todas as outras permanecem em ALTO. Isto será indicado pela presença de pequenos círculos nas linhas de saída no diagrama do decodificador.

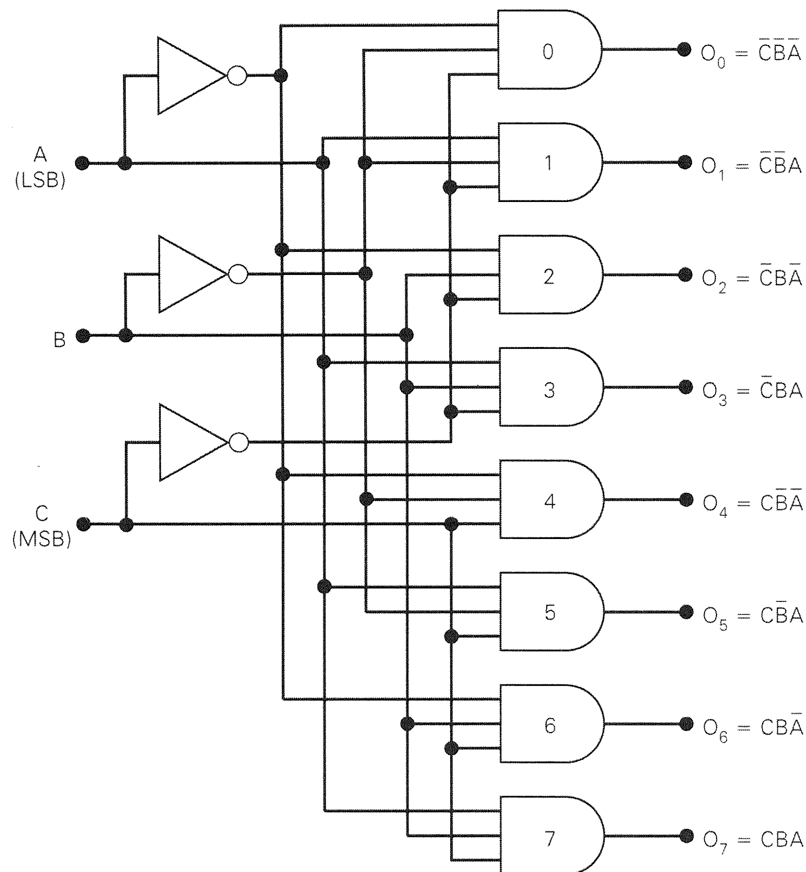
Alguns decodificadores não utilizam todos os 2^N códigos de entrada possíveis, mas apenas alguns. Por exemplo, um decodificador BCD para decimal tem um código de entrada de quatro bits e *dez* linhas de saída, que correspondem aos *dez* códigos BCD de 0000 até 1001. Decodificadores deste tipo freqüentemente são projetados, de modo que caso um código não-usado seja aplicado na entrada, *nenhuma* saída será ativada.

No Cap. 7 estudamos como decodificadores são usados com contadores para detectar os diversos estágios de contagem. Nesse tipo de aplicação, os FFs do contador forneciam os códigos binários de entrada para o decodificador. O mesmo circuito decodificador básico é usado, não importando de onde vêm as entradas. A Fig. 9-2 mostra o circuito para um decodificador de três entradas e $2^3 = 8$ saídas. Ele utiliza somente portas AND, e portanto as saídas são ativas em ALTO. Note que, para um determinado código de entrada, a única saída que está ativa (ALTO) é aquela que corresponde ao equivalente decimal do código binário de entrada (isto é, a saída O_6 vai para ALTO somente quando $CBA = 110_2 = 6_{10}$).

Este decodificador pode ser identificado de várias maneiras. Ele pode ser chamado *decodificador de 3 linhas para 8 linhas*, porque tem três linhas de entrada e oito linhas de saída. Também pode ser chamado de *decodificador* ou *conversor binário para octal*, porque recebe um código binário de entrada de três bits e ativa uma dentre as oito (octal) saídas correspondente para aquele código. Ele também é denominado de *decodificador 1 de 8*, porque somente 1 das 8 saídas é ativada de cada vez.

Entradas de HABILITAÇÃO

Alguns decodificadores têm uma ou mais entradas de HABILITAÇÃO (ENABLE), que são usadas para controlar a operação do decodificador. Por exemplo, consulte o decodificador da Fig. 9-2 e imagine que ele tem uma linha de HABILITAÇÃO comum conectada numa quarta entrada de cada porta. Com esta linha de HABILITAÇÃO mantida



C	B	A	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

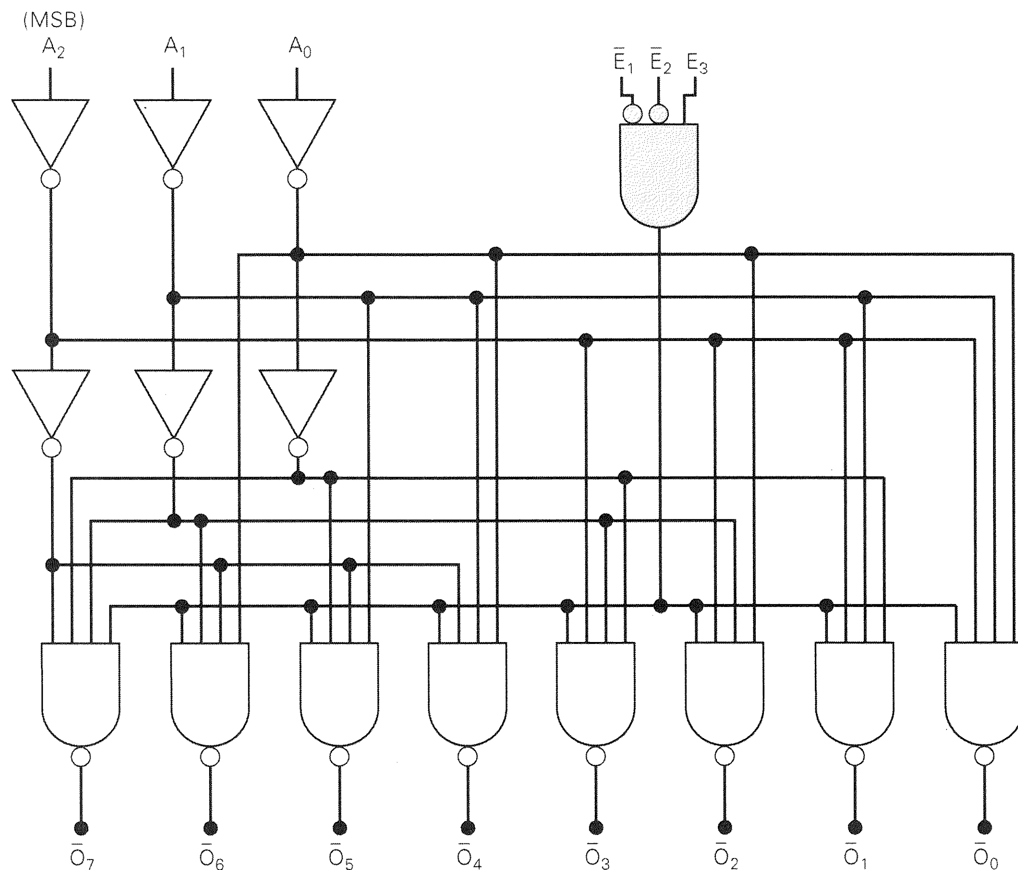
Fig. 9-2 Decodificador de 3 linhas para 8 linhas (ou 1 de 8).

em ALTO, o decodificador funciona normalmente, e o código de entrada, *A*, *B* e *C*, determina qual das saídas fica em ALTO. Entretanto, com a HABILITAÇÃO mantida em BAIXO, todas as saídas são forçadas para o estado BAIXO, não importando os níveis nas entradas *A*, *B* e *C*. Assim, o decodificador é habilitado somente quando HABILITAÇÃO está em ALTO.

A Fig. 9-3(a) mostra o diagrama lógico para o decodificador 74LS138, conforme ele aparece no *Manual TTL (TTL Data Book)* da Fairchild. Examinando este diagrama cuidadosamente, podemos determinar exatamente como este decodificador funciona. Primeiramente, note que ele tem saídas com portas NAND, de modo que suas saídas são ativas em BAIXO. Outra indicação é a identificação das saídas como $\overline{O}_7, \overline{O}_6, \overline{O}_5$ e assim por diante; a barra indica saídas ativas em BAIXO.

O código de entrada é aplicado em A_2, A_1 e A_0, onde A_2 é o MSB. Com três entradas e oito saídas, este é um decodificador 3 para 8 ou, de modo equivalente, um decodificador 1 de 8.

As entradas $\overline{E}_1$, $\overline{E}_2$ e E_3 são entradas de habilitação separadas que são combinadas na porta AND. Para habilitar as portas NAND de saída para responder ao código de entrada em $A_2A_1A_0$, a saída desta porta AND deve estar em ALTO. Isto só ocorre quando $\overline{E}_1 = \overline{E}_2 = 0$ e $E_3 = 1$. Em outras palavras, $\overline{E}_1$ e $\overline{E}_2$ são ativos em BAIXO, E_3 é ativo em ALTO, e todos três devem estar ativos para habilitar as saídas do decodificador. Se uma ou mais entradas de habilitação estão em seu estado inativo, a saída da porta AND fica em BAIXO, o que força todas as saídas NAND para seus respectivos estados inativos em ALTO, não importando o código de entrada. Esta operação está resumida na tabela-



$\overline{E}_1$	$\overline{E}_2$	E_3	Saídas
0	0	1	Responde ao código de entrada $A_2A_1A_0$
1	X	X	Desabilitadas — todas em ALTO
X	1	X	Desabilitadas — todas em ALTO
X	X	0	Desabilitadas — todas em ALTO

(b)

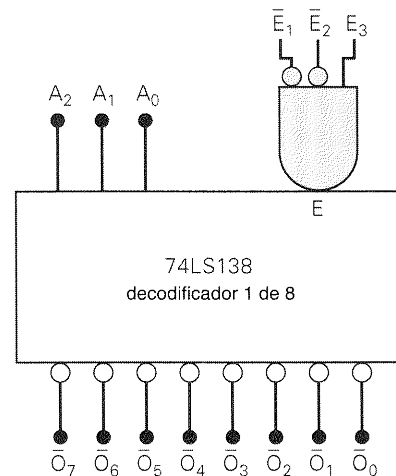


Fig. 9-3 (a) Diagrama lógico para o decodificador 74LS138; (b) tabela-verdade; (c) símbolo lógico. (Cortesia da Fairchild, uma empresa da Schlumberger.)

verdade da Fig. 9-3(b). Lembre-se de que x representa a condição don't care.

O símbolo lógico para o 74LS138 é mostrado na Fig. 9-3(c). Note como as saídas ativas em BAIXO e como as entradas de habilitação são representadas. Embora a porta AND de habilitação esteja mostrada externamente ao bloco decodificador, ela é parte do circuito interno do CI. O 74HC138 é a versão CMOS de alta velocidade deste decodificador.

EXEMPLO 9-1

Indique os estados das saídas do 74LS138 para cada um dos seguintes conjuntos de entrada.

(a) $E_3 = \overline{E}_2 = 1$, $\overline{E}_1 = 0$, $A_2 = A_1 = 1$, $A_0 = 0$

(b) $E_3 = 1$, $\overline{E}_2 = \overline{E}_1 = 0$, $A_2 = 0$, $A_1 = A_0 = 1$

Solução

(a) Com $\overline{E}_2 = 1$, o decodificador está desabilitado e todas as saídas estarão no estado inativo ALTO. Isto pode ser determinado a partir da tabela-verdade ou seguindo os níveis de entrada através do circuito lógico.

(b) Todas as entradas de habilitação estão ativadas, portanto a área de decodificação está habilitada. Ela decodifica o código de entrada $011_2 = 3_{10}$ para ativar a saída $\overline{O}_3$. Logo, $\overline{O}_3$ fica em BAIXO e todas as outras saídas ficam em ALTO.

EXEMPLO 9-2

A Fig. 9-4 mostra como quatro 74LS138s e um INVERSOR podem ser interligados para funcionarem como um decodificador 1 de 32. Os decodificadores foram identificados de Z_1 até Z_4 para facilitar a consulta, e as oito saídas de cada um são combinadas em 32 saídas. As saídas de Z_1 são $\overline{O}_0$ a $\overline{O}_7$; as saídas de Z_2 foram renomeadas para $\overline{O}_8$ a $\overline{O}_{15}$, respectivamente; as saídas de Z_3 foram renomeadas para $\overline{O}_{16}$ a $\overline{O}_{23}$; e as saídas de Z_4 foram renomeadas para $\overline{O}_{24}$ a $\overline{O}_{31}$. Um código de entrada de cinco bits, $A_4A_3A_2A_1A_0$, ativa apenas uma destas 32 saídas para cada um dos 32 códigos de entrada possíveis.

(a) Qual saída será ativada para $A_4A_3A_2A_1A_0 = 01101$?

(b) Que faixa de códigos de entrada ativará o chip Z_4?

Solução

(a) O código de cinco bits tem duas partes distintas. Os bits A_4 e A_3 determinam qual dos chips decodificadores, Z_1 a Z_4, estará habilitado, enquanto $A_2A_1A_0$ determinam qual saída do chip habilitado será ativada. Com $A_4A_3 = 01$, apenas Z_2 tem todas as suas entradas de habilitação ativadas. Logo, Z_2 responde ao código $A_2A_1A_0 = 101$ e ativa sua saída $\overline{O}_5$, que foi renomeada para $\overline{O}_{13}$. Assim, o código de entrada 01101, que é o equivalente binário do decimal 13, faz com que a saída $\overline{O}_{13}$ vá para BAIXO, enquanto todas as outras permanecem em ALTO.

(b) Para habilitar Z_4, tanto A_4 como A_3 devem estar em ALTO. Logo, todos os códigos de entrada que variam de 11000 (24_{10}) até 11111 (31_{10}) ativarão Z_4. Isto corresponde às saídas $\overline{O}_{24}$ a $\overline{O}_{31}$.

Decodificadores BCD para Decimal

A Fig. 9-5(a) mostra o diagrama lógico para o **decodificador BCD para decimal** 7442. Ele também está disponível como um 74LS42 ou um 74HC42. Cada saída vai para BAIXO apenas quando sua entrada BCD correspondente é aplicada. Por exemplo, $\overline{O}_5$ vai para BAIXO somente com as entradas $DCBA = 0101$; $\overline{O}_8$ vai para BAIXO somente com $DCBA = 1000$. Para combinações de entrada que são inválidas para BCD, nenhuma das saídas será ativada. Este decodificador pode também ser denominado *decodificador 4 para 10* ou um *decodificador 1 de 10*. O símbolo lógico e a tabela-verdade para o 7442 também são mostrados na figura. Note que este decodificador não tem uma entrada de habilitação. No Problema 9-7 vamos ver como o 7442 pode ser utilizado como um decodificador 3 para 8 com a entrada D sendo usada como entrada de habilitação.

Decodificador/Driver BCD para Decimal

O chip TTL 7445 é um decodificador/**driver** BCD para decimal. O termo *driver* é adicionado na sua descrição porque este CI tem saídas em coletor aberto que podem ope-

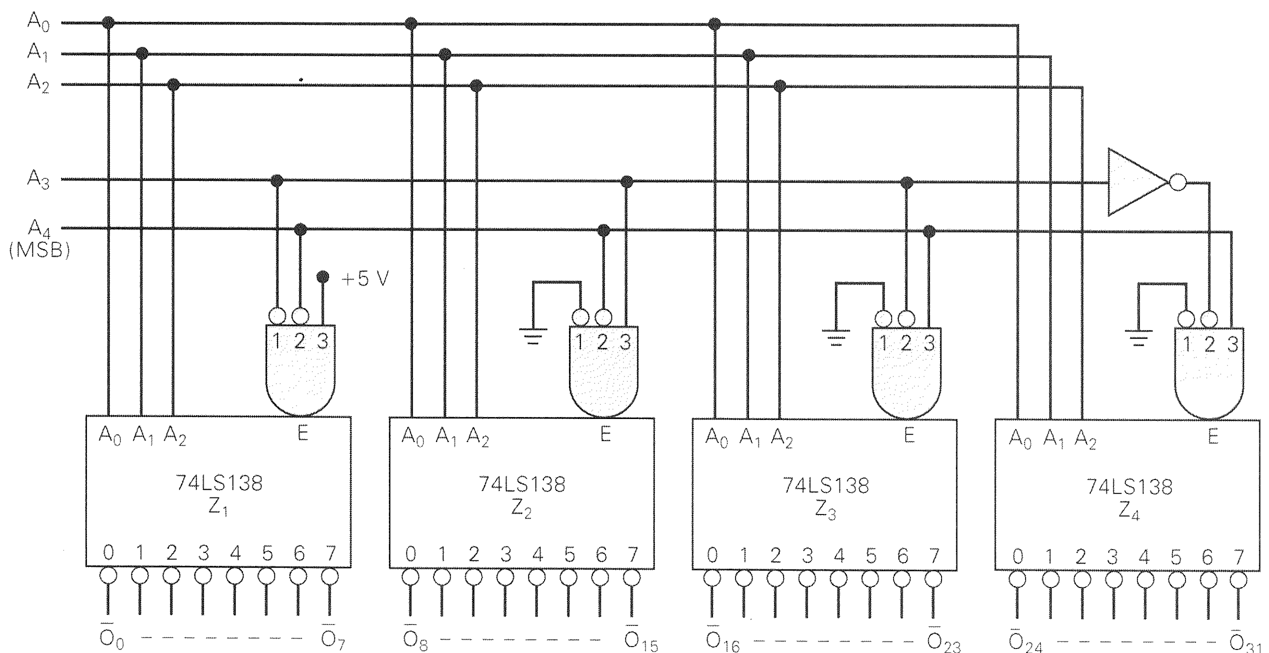
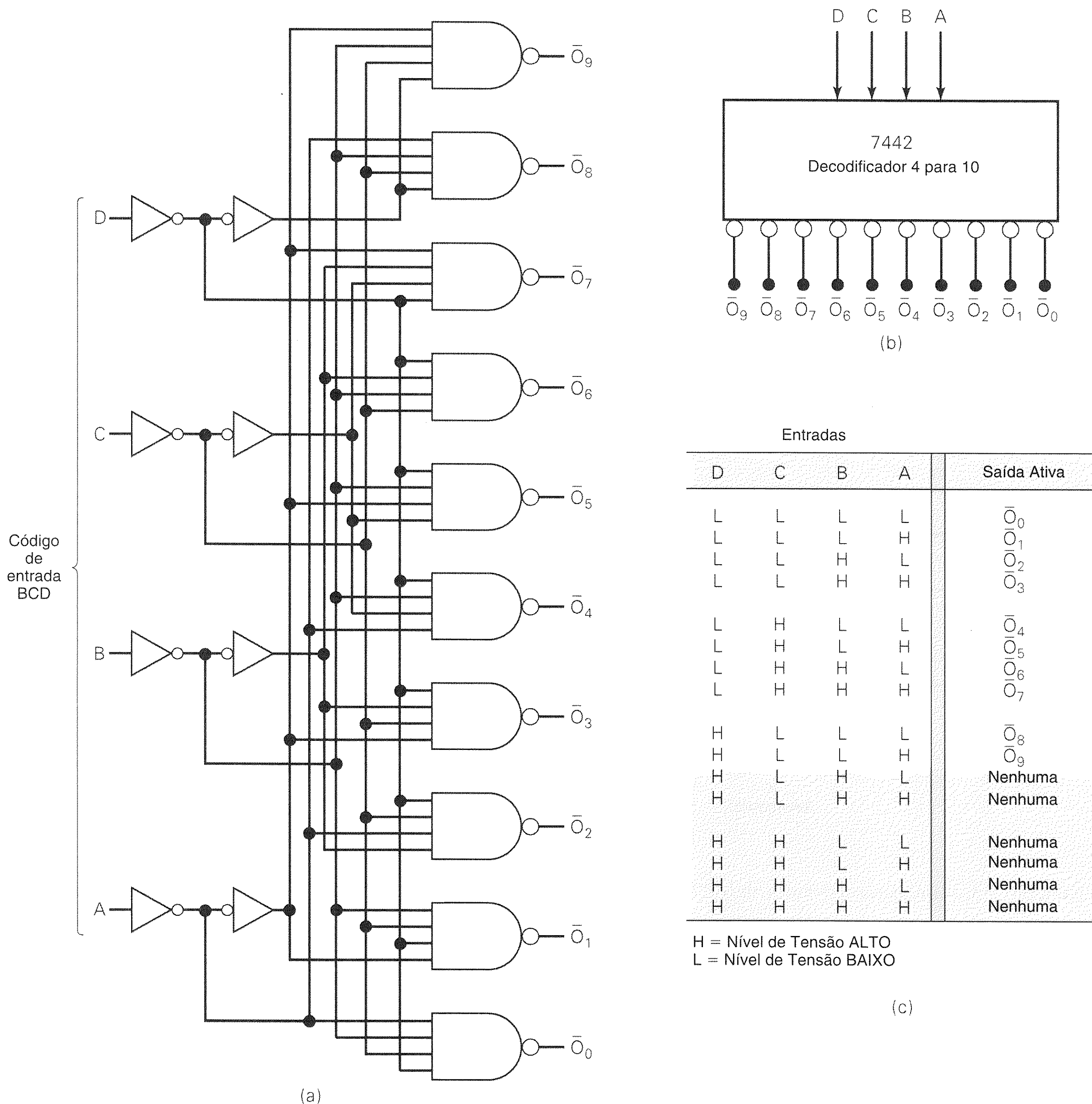


Fig. 9-4 Quatro 74LS138s formando um decodificador 5 para 32 (ou 1 de 32).



D	C	B	A	Saída Ativa
L	L	L	L	$\bar{O}_0$
L	L	L	H	$\bar{O}_1$
L	L	H	L	$\bar{O}_2$
L	L	H	H	$\bar{O}_3$
L	H	L	L	$\bar{O}_4$
L	H	L	H	$\bar{O}_5$
L	H	H	L	$\bar{O}_6$
L	H	H	H	$\bar{O}_7$
H	L	L	L	$\bar{O}_8$
H	L	L	H	$\bar{O}_9$
H	L	H	L	Nenhuma
H	L	H	H	Nenhuma
H	H	L	L	Nenhuma
H	H	L	H	Nenhuma
H	H	H	L	Nenhuma
H	H	H	H	Nenhuma

H = Nível de Tensão ALTO
L = Nível de Tensão BAIXO

Fig. 9-5 (a) Diagrama lógico para o decodificador BCD para decimal 7442; (b) símbolo lógico; (c) tabela-verdade. (Cortesia da Fairchild, uma empresa da Schlumberger.)

rar com tensões e correntes mais altas do que os limites de uma saída normal TTL. As saídas do 7445 podem absorver até 80 mA no estado BAIXO e podem ser levadas a até 30 V no estado ALTO. Isto faz com que elas sejam apropriadas para acionar diretamente cargas tais como lâmpadas e LEDs indicadores, relés ou motores dc.

Aplicações de Decodificadores

Decodificadores são usados sempre que uma saída ou grupo de saídas deve ser ativado somente na ocorrência de uma combinação específica de níveis de entrada. Estes níveis de entrada são freqüentemente fornecidos pelas saídas de um contador ou de um registrador. Quando as entradas do decodificador vêm de um contador que está sendo acionado continuamente, as saídas do decodificador serão ativadas seqüencialmente, e elas podem ser utilizadas como sinais de temporização ou seqüenciamento para ligar ou desligar dispositivos em determinados momentos. Um exemplo desta operação é mostrado na Fig. 9-6 usando o contador 74LS293 e o decodificador/driver 7445 descrito anteriormente.

EXEMPLO 9-3

Descreva a operação do circuito na Fig. 9-6(a).

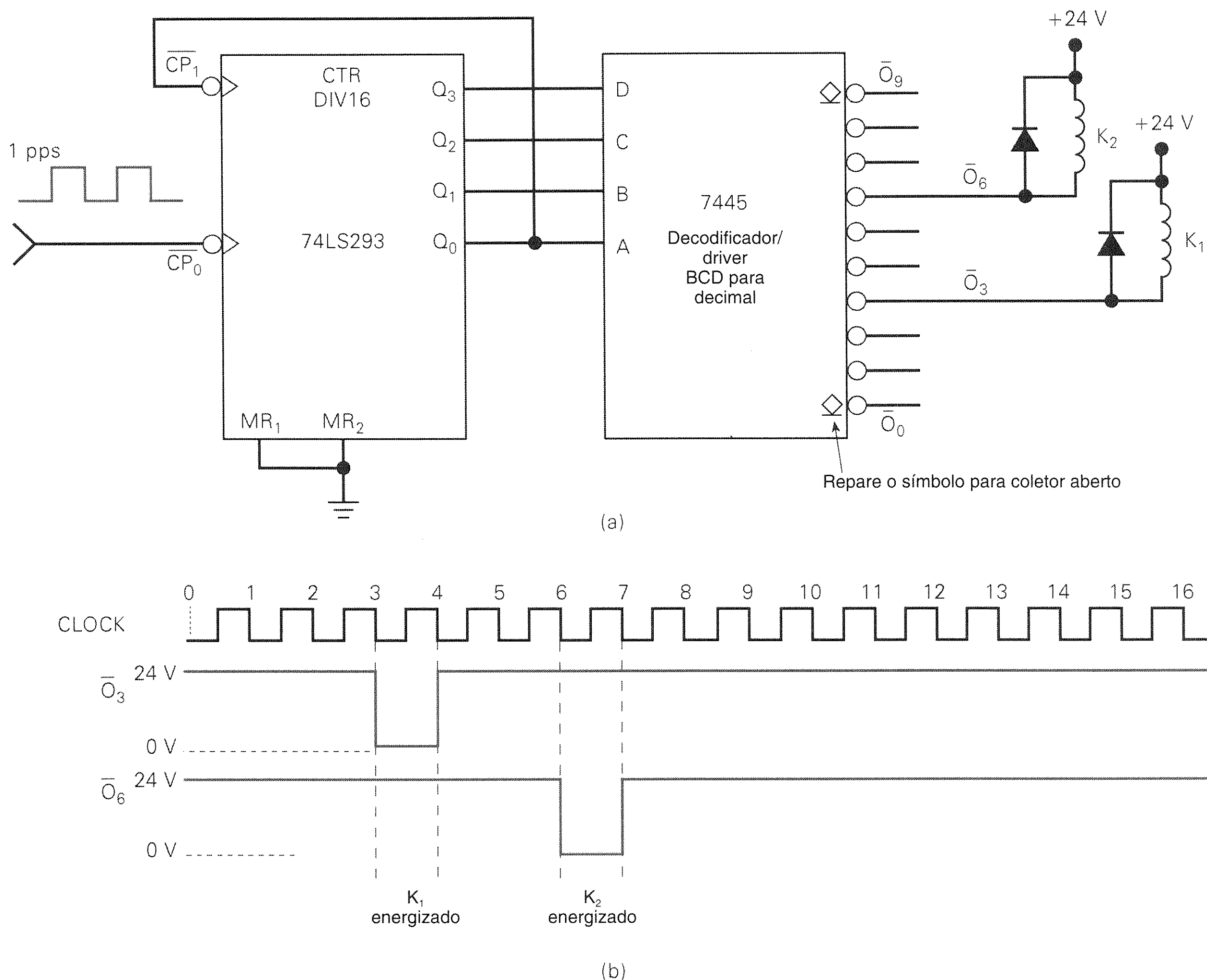


Fig. 9-6 Exemplo 9-3: combinação contador/decodificador utilizada para realizar operações de temporização e seqüenciamento.

Solução

O contador está sendo pulsado por um sinal de 1 pps e portanto realiza a contagem binária a uma taxa de 1 contagem/s. As saídas do contador estão conectadas nas entradas do decodificador. As saídas em coletor aberto $\overline{O}_3$ e $\overline{O}_6$ do 7445 são usadas para ligar e desligar os relés K_1 e K_2. Por exemplo, quando $\overline{O}_3$ estiver no seu estado inativo ALTO, seu transistor de saída estará cortado (não conduzindo), de modo que nenhuma corrente poderá fluir através do relé K_1 e ele estará desenergizado. Quando $\overline{O}_3$ estiver no seu estado ativo em BAIXO, seu transistor de saída estará ligado e atuará de maneira a absorver a corrente através de K_1, de modo que K_1 estará energizado. Note que os relés operam com +24 V. Repare também a presença dos diodos ligados nas bobinas dos relés; eles protegem os transistores de saída do decodificador do grande "impulso indutivo" de tensão que seria produzido quando a corrente na bobina fosse interrompida abruptamente.

O diagrama de tempo na Fig. 9-6(b) mostra a seqüência de eventos. Se admitimos que o contador está no estado 0000 no instante 0, então as saídas $\overline{O}_3$ e $\overline{O}_6$ estão inicialmente inativas no estado ALTO, onde seus transistores de saída estão desligados e ambos os relés desenergizados. Conforme os pulsos de clock vão sendo aplicados, o contador é incrementado uma vez a cada segundo. Na descida do terceiro pulso (instante de tempo 3), o contador vai para o estado 0011 (3). Isto ativa a saída $\overline{O}_3$ do decodificador que por sua vez energiza K_1. Na descida do quarto pulso, o contador vai para o estado 0100 (4). Isto desativa $\overline{O}_3$ e desenergiza o relé K_1.

Analogamente, no instante 6 o contador vai para o estado 0110 (6); isto faz $\overline{O}_6 = 0$ e energiza K_2. No instante 7, o contador vai para 0111 (7) e desativa O_6 para desenergizar K_2.

O contador continuará a contar os pulsos aplicados. Após 16 pulsos, a seqüência que acabamos de descrever será reiniciada.

Os decodificadores são amplamente utilizados nos sistemas de memória dos computadores, onde respondem aos endereços gerados pelo processador central para acessar uma posição de memória específica. Cada CI de memória contém muitos registradores que podem armazenar números binários (dados). Cada registrador precisa ter seu pró-

prio endereço único, para distingui-lo de todos os outros registradores. Um decodificador é construído nos circuitos internos dos CIs de memória e permite que um determinado registrador de armazenamento seja ativado quando uma única combinação de entrada (isto é, seu endereço) é aplicada. Em um sistema existem usualmente diversos CIs de memória combinados para implementar a capacidade de armazenamento completa. Um decodificador é usado para selecionar um chip de memória, em resposta a uma faixa de endereços, decodificando os bits mais significativos de endereço do sistema e habilitando (selecionando) um determinado chip. Examinaremos esta aplicação no Problema 9-60 e com mais profundidade quando estudarmos memórias no Cap. 11.

Em sistemas de memória mais complicados, os chips de memória são organizados em múltiplos bancos, os quais devem ser selecionados individualmente ou simultaneamente, dependendo de se o microprocessador deseja um ou mais bytes por vez. Isto significa que, sob certas circunstâncias, mais do que uma saída do decodificador deve ser ativada. Para sistemas deste tipo, um dispositivo de lógica programável freqüentemente é usado para implementar o decodificador, pois um simples decodificador 1 de 8 sozinho não é suficiente. Dispositivos de lógica programável podem ser facilmente utilizados para aplicações específicas de decodificação.

Questões de Revisão

1. Mais de uma saída de um decodificador pode ser ativada de cada vez?
2. Qual é a função da(s) entrada(s) de habilitação de um decodificador?
3. Em que um 7445 difere de um 7442?
4. O 74154 é um decodificador de 4 para 16 com duas entradas de habilitação ativas em BAIXO. Quantos pinos (incluindo alimentação e terra) este CI tem?

9-2 DECODIFICADORES/DRIVERS BCD PARA 7 SEGMENTOS

A maioria dos equipamentos digitais tem algum meio para mostrar as informações num formato, que pode ser prontamente compreendido pelo usuário ou operador. Estas informações freqüentemente são dados numéricos, mas também podem ser alfanuméricos (números e letras). Um dos métodos mais simples e populares para a apresentação de dígitos numéricos usa uma configuração de 7 segmentos [Fig. 9-7(a)] para formar os caracteres decimais de 0 a 9, e algumas vezes os caracteres hexadecimais de A até F. Um arranjo comum utiliza diodos emissores de luz (LEDs) para cada segmento. Controlando-se a corrente através de cada LED, alguns segmentos acendem e outros permanecem apagados, de modo que o padrão de caracter desejado seja gerado. A Fig. 9-7(b) mostra os padrões de segmentos usados para formar os vários dígitos. Por exemplo, para mostrar um "6", os segmentos *a, c, d, e, f* e *g* são acesos, enquanto o segmento *b* fica apagado.

Um **decodificador/driver BCD para 7 segmentos** é usado para receber uma entrada BCD de quatro bits e fornecer as saídas que acionam os segmentos apropriados para mostrar o dígito decimal. A lógica para este decodificador é mais complicada do que a lógica dos decodificadores que analisamos até agora, porque cada saída é acionada para mais do que uma combinação de entrada. Por exemplo, o segmento *e* deve ser ativado para qualquer um dos dígitos 0, 2, 6 e 8, o que significa sempre que qualquer um dos códigos 0000, 0010, 0110 ou 1000 ocorrer.

A Fig. 9-8(a) mostra um decodificador/driver BCD para 7 segmentos (TTL 7446 ou 7447) sendo usado para acionar um display a LEDs de 7 segmentos. Cada segmento consiste em um ou dois LEDs. Os anodos dos LEDs estão ligados em V_{CC} (+5 V). Os catodos dos LEDs são ligados através de resistores limitadores de corrente nas saídas apropriadas do decodificador/driver. O decodificador/driver tem saídas ativas em BAIXO, de coletor aberto, com transistores de acionamento que podem absorver uma corrente razoavelmente grande. Isto é necessário porque os displays a LED podem necessitar de 10 a 40 mA por segmento, dependendo do seu tipo e tamanho.

Para ilustrar a operação deste circuito, vamos supor que a entrada BCD seja $D = 0$, $C = 1$, $B = 0$, $A = 1$, que é 5 em BCD. Com estas entradas, as saídas $\overline{a}$, $\overline{f}$, $\overline{g}$, $\overline{c}$ e $\overline{d}$ do decodificador/driver ficariam acionadas em BAIXO (conectadas na terra), permitindo a corrente fluir através dos segmentos de LED *a, f, g, c* e *d*, e portanto apresentando o número 5. As saídas $\overline{b}$ e $\overline{e}$ estariam em ALTO (em aberto), de modo que os segmentos de LED *b* e *e* não acenderiam.

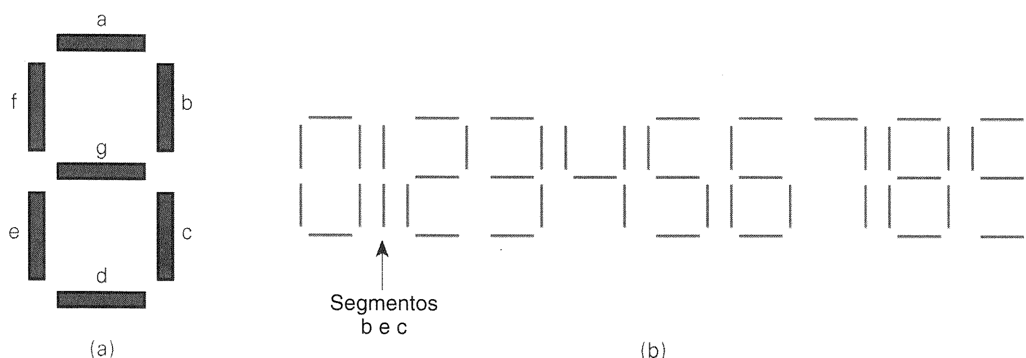


Fig. 9-7 (a) Configuração dos 7 segmentos; (b) segmentos ativos para cada dígito.

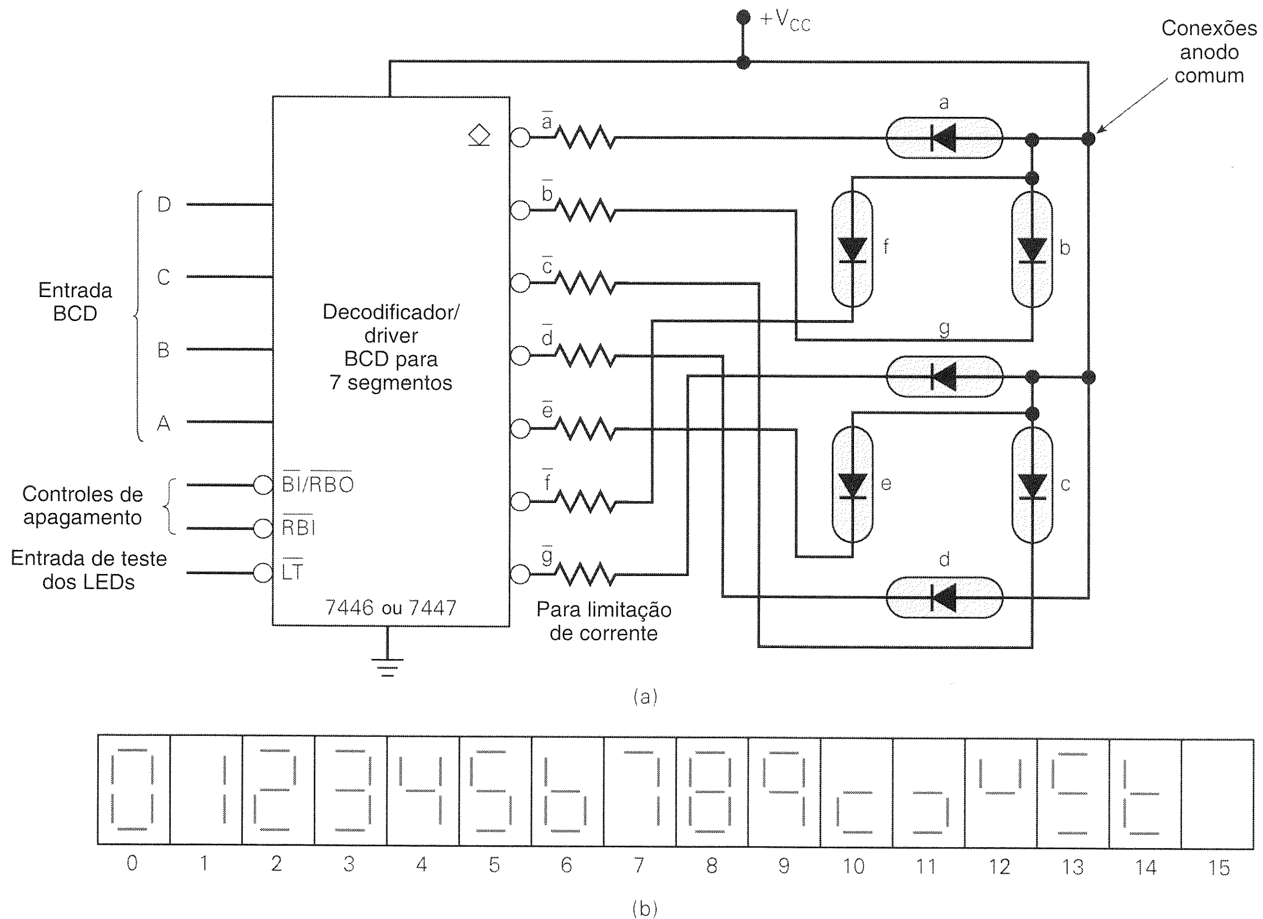


Fig. 9-8 (a) Decodificador/driver BCD para 7 segmentos acionando um display de 7 segmentos a LEDs de anodo comum; (b) padrões de segmentos para todos os códigos de entrada possíveis.

Os decodificadores/drivers 7446/47 são projetados para ativar segmentos específicos mesmo para códigos de entrada não-BCD (maiores do que 1001). A Fig. 9-8(b) mostra os padrões de segmentos ativados para todos os códigos de entrada possíveis desde 0000 até 1111. Note que um código de entrada 1111 (15) apaga todos os segmentos.

Decodificadores/drivers de 7 segmentos, tais como o 7446/47, são exceções para a regra de circuitos decodificadores ativarem apenas uma saída para cada combinação de entrada. Eles ativam um padrão único de saída para cada combinação de entrada.

Displays a LED de Catodo Comum versus Anodo Comum

O display a LEDs usado na Fig. 9-8 é do tipo **anodo comum** porque os anodos de todos os segmentos são conectados juntos em V_{CC}. Um outro tipo de display a LEDs de 7 segmentos utiliza um arranjo com **catodo comum**, onde os catodos de todos os segmentos são conectados juntos na terra. Este tipo de display deve ser acionado por um decodificador/driver BCD para 7 segmentos com saídas ativas em ALTO, que aplicam uma tensão nos anodos daqueles segmentos que devem ser ativados. Tendo em vista que cada segmento necessita de uma corrente de 10 a 20 mA para acender, dispositivos TTL e CMOS não são normalmente utilizados para acionar diretamente displays de catodo comum. Lembre-se, do Cap. 8, de que saídas TTL e CMOS não são capazes de fornecer uma grande corrente. Um circuito de interface com transistores é freqüentemente usado entre os chips de decodificação e o display de catodo comum.

EXEMPLO 9-4

Cada segmento de um típico display a LEDs de 7 segmentos opera com 10 mA e 2,7 V para um brilho normal. Calcule o valor do resistor limitador de corrente necessário para produzir aproximadamente 10 mA por segmento.

Solução

Consultando a Fig. 9-8(a), podemos verificar que o resistor em série deve ter uma queda de tensão igual à diferença entre V_{CC} = 5 V e a tensão de 2,7 V. Estes 2,3 V sobre o resistor devem produzir uma corrente de aproximadamente 10 mA. Logo, temos

$$R_S = \frac{2{,}3\ \text{V}}{10\ \text{mA}} = 230\ \Omega$$

Um valor padrão de resistor próximo a este pode ser usado. Um resistor de 220 Ω seria uma boa escolha.

Questões de Revisão

1. Quais os segmentos de LEDs que estarão ligados para uma entrada de 1001 num decodificador/driver?
2. *Verdadeiro ou falso:* Mais de uma saída de um decodificador/driver BCD para 7 segmentos pode estar ativa de uma vez.

9-3 DISPLAYS DE CRISTAL LÍQUIDO

Um display a LEDs gera ou emite energia luminosa conforme a corrente passa por cada segmento. Um display de cristal líquido (**LCD** — *liquid crystal display*) controla a reflexão da luz disponível. A luz disponível pode ser simplesmente a luz ambiente, tal como a luz do sol ou a iluminação normal; LCDs *reflexivos* utilizam luz ambiente. A luz também pode ser fornecida por uma pequena fonte luminosa que faz parte da unidade de display; LCDs *backlit* utilizam este método. Em qualquer dos casos, os LCDs obtiveram ampla aceitação devido ao baixíssimo consumo de energia quando comparado aos LEDs, especialmente em equipamentos que operam com baterias, tais como calculadoras, relógios digitais e instrumentos eletrônicos portáteis de medição. Os LEDs têm a vantagem de proporcionarem um display com brilho muito mais intenso, que, ao contrário dos LCDs reflexivos, são facilmente visíveis em áreas escuras ou pouco iluminadas.

Basicamente, os LCDs operam com sinais AC de baixa tensão (tipicamente de 3 a 15 V rms) e baixa freqüência (25 a 60 Hz) e consomem uma corrente muito pequena. Eles freqüentemente são configurados como displays de 7 segmentos para leituras numéricas conforme mostrado na Fig. 9-9(a). A tensão AC necessária para ligar um segmento é aplicada entre o segmento e o **backplane**, que é comum a todos os segmentos. O segmento e o backplane formam um capacitor que consome uma corrente muito baixa enquanto a freqüência AC é mantida baixa. Ela geralmente não é inferior a 25 Hz, porque poderia produzir uma cintilação visível.

Uma explicação, certamente simplificada, de como um LCD funciona seria a seguinte. Quando não há diferença de tensão entre um segmento e o backplane, diz-se que o segmento está *desativado* (OFF). Os segmentos *d, e, f* e *g*, na Fig. 9-9(b), estão OFF e refletirão a luz incidente, de modo que parecerão invisíveis contra o fundo. Quando uma tensão AC apropriada é aplicada entre um segmento e o backplane, o segmento é ativado (ON). Os segmentos *a, b* e *c*, na Fig. 9-9(b), estão ON e não refletirão a luz incidente, e portanto parecerão escuros contra o fundo.

Acionando um LCD

Um segmento de LCD ligará quando uma tensão AC for aplicada entre o segmento e o backplane, e desligará quando não existir tensão entre os dois. Em vez de gerar um sinal AC, é prática comum produzir a tensão AC necessária aplicando-se ondas quadradas fora de fase ao segmento e ao backplane. Isto está ilustrado na Fig. 9-10 para um segmento. Uma onda quadrada de 40 Hz é aplicada ao backplane e também na entrada de uma EX-OR CMOS 74HC86. A outra entrada da EX-OR é uma entrada de CONTROLE, que controla se o segmento está ligado ou desligado.

Quando a entrada de CONTROLE estiver em BAIXO, a saída da EX-OR será exatamente a mesma onda quadrada de 40 Hz, de modo que os sinais aplicados ao segmento e ao backplane serão iguais. Como não existe diferença de tensão, o segmento estará desligado. Quando a entrada de CONTROLE estiver em ALTO, a saída da EX-OR será o INVERSO da onda quadrada de 40 Hz, de modo que o sinal aplicado ao segmento estará fora de fase com o sinal aplicado ao backplane. Como resultado, a tensão do segmento estará alternadamente em +5 V e em −5 V em relação ao backplane. Esta tensão AC ligará o segmento.

A mesma idéia pode ser estendida para um display LCD de 7 segmentos completo conforme mostrado na Fig. 9-11. Nesta figura, o decodificador/driver CMOS BCD para 7 segmentos 74HC4511 fornece os sinais de CONTROLE para cada uma das sete EX-ORs para os sete segmentos. O 74HC4511 tem saídas ativas em ALTO, já que um nível ALTO é necessário para ligar um segmento. O decodificador/driver e as portas EX-OR da Fig. 9-11 estão disponíveis em um único chip. Tal dispositivo é o chip CMOS 74HC4543. Ele recebe o código BCD de entrada e fornece

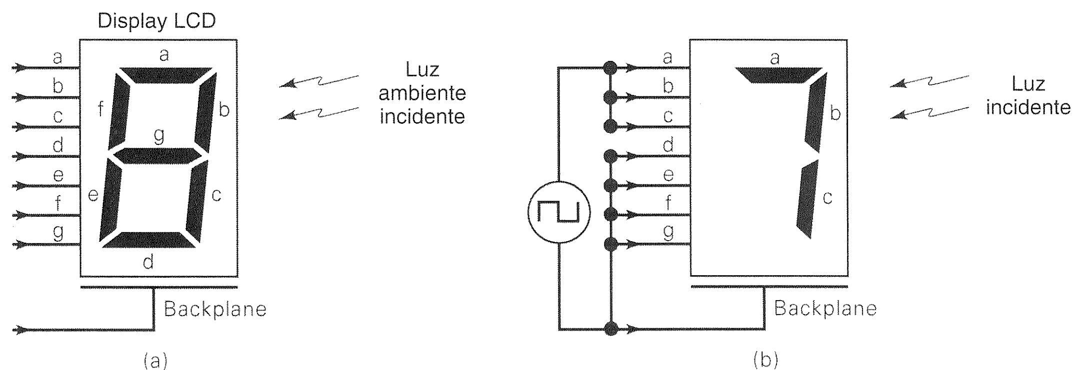


Fig. 9-9 Display de cristal líquido: (a) configuração básica; (b) aplicando-se uma tensão entre o segmento e o backplane, o segmento é ligado. Uma tensão zero desliga o segmento.

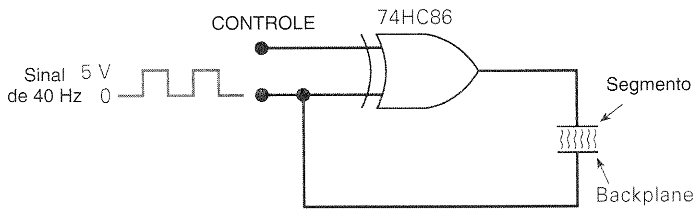


Fig. 9-10 Método para acionar um segmento de LCD. Quando CONTROLE está em BAIXO, o segmento está desligado. Quando CONTROLE está em ALTO, o segmento está ligado.

as saídas necessárias para acionar diretamente os segmentos LCD.

De um modo geral, dispositivos CMOS são usados para acionar LCDs por duas razões: (1) eles necessitam de muito menos potência do que os TTL e são mais adequados para aplicações operadas com bateria onde os LCDs são usados; (2) o estado BAIXO TTL não é exatamente 0 V e pode ser até 0,4 V. Isto produziria um componente de tensão DC entre segmento e backplane que encurtaria consideravelmente a vida de um LCD.

Tipos de LCDs

Cristais líquidos estão disponíveis como displays numéricos decimais de vários dígitos. Eles são oferecidos em muitos tamanhos e com diversos caracteres especiais, tais como dois pontos (:) para displays de relógios; indicadores +/− para voltímetros digitais; vírgula decimal para calculadoras; e indicadores do estado da bateria, tendo em vista que muitos dispositivos com LCD são alimentados por bateria. Estes displays devem ser acionados por um chip decodificador/driver tal como o 74HC4543 ou o 74C945.

Um display LCD mais complicado, mas prontamente disponível, é o módulo LCD alfanumérico. Ele é disponibilizado por diversas companhias em vários formatos, tais como o de uma linha com 16 caracteres ou o de 4 linhas de 40 caracteres. A interface com estes módulos foi padronizada, de modo que um módulo LCD de um fabricante utiliza os mesmos sinais e formatos de dados. O módulo inclui alguns chips VLSI que tornam o dispositivo simples de usar. Oito linhas de dados são utilizadas para enviar o código ASCII do que você deseja mostrar no display. Estas linhas também recebem códigos de controle especiais para o registrador de comandos do LCD. Três outras entradas (Register Select, Read/Write e Enable) são usadas para controlar a localização, a direção e a temporização da transferência de dados. Conforme os caracteres vão sendo enviados para o módulo, ele os armazena em sua própria memória e apresenta-os na tela do display.

Outros módulos LCDs permitem ao usuário criar um display gráfico, controlando os pontos individuais da tela chamados **pixels**. Grandes painéis LCD podem ser atualizados a uma taxa elevada, produzindo filmes de alta qualidade. Nestes displays, as linhas de controle são organizadas em uma ma-

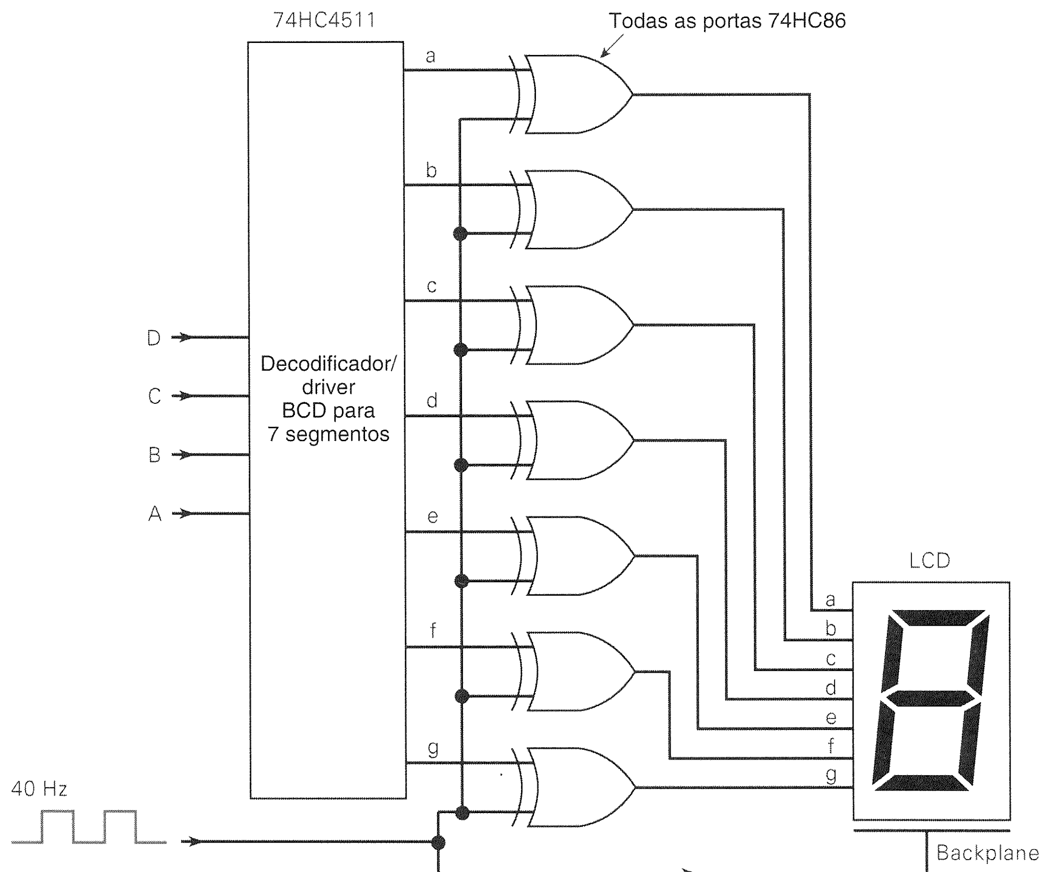


Fig. 9-11 Método para acionamento de um LCD de 7 segmentos.

lha de linhas e colunas. Na interseção de cada linha e coluna está um pixel que atua como uma "janela", que pode ser eletronicamente aberta e fechada para controlar a quantidade de luz que é transmitida através da célula. A tensão entre a linha e a coluna determina o brilho de cada pixel. Num computador laptop, um número binário para cada pixel é armazenado na memória de "vídeo". Estes números são convertidos para tensões que são aplicadas ao display.

Cada pixel em um display colorido é formado na verdade por três subpixels. Estes subpixels controlam a luz que passa através dos filtros vermelho, verde e azul para produzir a cor de cada pixel. Em uma tela LCD de 640 por 480 existiriam 640 × 3 conexões para colunas e 480 conexões para linhas, para um total de 2400 conexões para o LCD. Obviamente, o circuito de acionamento para tal dispositivo é um circuito VLSI muito complicado que é fabricado como uma parte do display.

Displays a Gás

Uma outra tecnologia popular para displays numéricos digitais é o **display a gás**. Estes displays são invólucros de vidro selados a vácuo, que são carregados com gás neon que brilha na vizinhança de um eletrodo dentro do invólucro de vidro. Os eletrodos são configurados no padrão familiar de 7 segmentos, numa matriz de pontos de 5 por 7. Uma tensão relativamente alta (por volta de 100 V) é aplicada entre um eletrodo de segmento e um eletrodo de referência que está selado dentro do display, mas que usualmente não é visível ao usuário. Os circuitos de acionamento precisam levar o segmento para o nível de terra e absorver uma pequena quantidade de corrente que flui. Eles também devem ser capazes de manipular altos potenciais em suas saídas quando estão em ALTO. Estes displays têm a vantagem de alta visibilidade no escuro, além de grandes segmentos, que consomem menos corrente que os displays a LED.

Questões de Revisão

1. Indique quais das seguintes declarações se referem a displays LCD e quais se referem a displays a LEDs.
   (a) Emite luz.
   (b) Reflete a luz ambiente.
   (c) São melhores para aplicações de baixa potência.
   (d) Necessitam de uma tensão AC.
   (e) Utilizam uma configuração de 7 segmentos para produzirem dígitos.
   (f) Necessitam de resistores limitadores de corrente.
2. Que tipo de dado é enviado para cada um dos seguintes itens:
   (a) Um display LCD de 7 segmentos com um decodificador/driver
   (b) Um módulo LCD alfanumérico
   (c) Um display LCD de computador

9-4 CODIFICADORES

A maioria dos decodificadores aceita um código de entrada e produz um nível ALTO (ou BAIXO) em *uma e somente uma* linha de saída. Em outras palavras, podemos dizer que

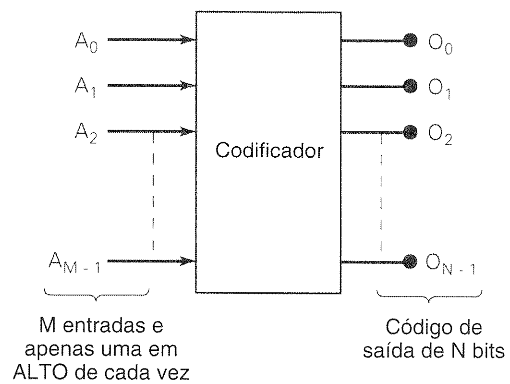


Fig. 9-12 Diagrama geral do codificador.

um decodificador identifica, reconhece ou detecta um código específico. O oposto deste processo de decodificação é chamado **codificação** e é realizado por um circuito lógico denominado **codificador**. Um codificador tem um certo número de linhas de entrada, onde somente uma delas é ativada por vez, e produz um código de saída de N bits, dependendo de qual entrada está ativada. A Fig. 9-12 é o diagrama geral para um codificador com M entradas e N saídas. Neste caso as entradas são ativas em ALTO, o que significa que estão normalmente em BAIXO.

Vimos que um *decodificador binário para octal (decodificador 3 linhas para 8 linhas)* aceita um código de entrada de três bits e ativa uma dentre oito linhas de saída correspondente a este código. Um *codificador octal para binário (codificador de 8 linhas para 3 linhas)* realiza a função oposta: ele aceita oito linhas de entrada e produz um código de saída de três bits correspondente a entrada ativada. A Fig. 9-13 mostra o circuito lógico e a tabela-verdade para um codificador octal para binário com entradas ativas em BAIXO.

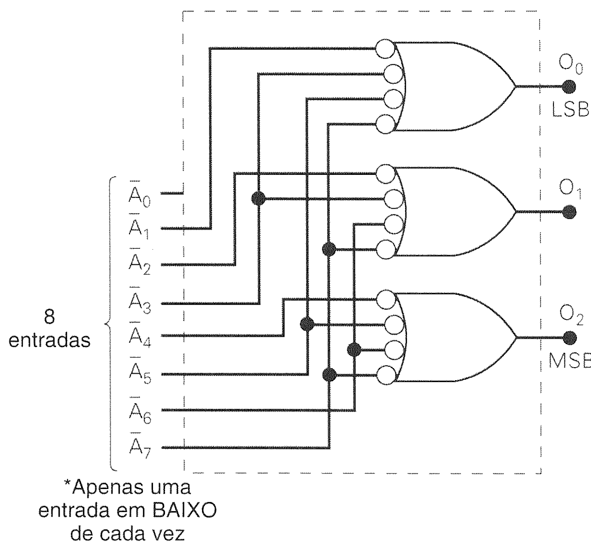
Seguindo a lógica, podemos verificar que um nível BAIXO em qualquer uma das entradas de cada vez produzirá o código binário de saída correspondente para aquela entrada. Por exemplo, um nível BAIXO em $\overline{A}_3$ (enquanto todas as outras entradas estiverem em ALTO) produzirá $O_2 = 0$, $O_1 = 1$ e $O_0 = 1$, que é o código binário para 3. Note que $\overline{A}_0$ não está conectado nas portas lógicas, pois as saídas do codificador normalmente estarão em 000 quando nenhuma das entradas de $\overline{A}_1$ até $\overline{A}_9$ estiver em BAIXO.

EXEMPLO 9-5

Determine as saídas do codificador na Fig. 9-13 quando $\overline{A}_3$ e $\overline{A}_5$ estiverem simultaneamente em BAIXO.

Solução

Seguindo através das portas lógicas, vemos que os níveis BAIXOs nestas duas entradas produzirão níveis ALTOs em cada saída, em outras palavras, o código binário 111. Certamente, este não é o código de nenhuma das entradas ativadas.



Entradas								Saídas		
$\overline{A_0}$	$\overline{A_1}$	$\overline{A_2}$	$\overline{A_3}$	$\overline{A_4}$	$\overline{A_5}$	$\overline{A_6}$	$\overline{A_7}$	O_2	O_1	O_0
X	1	1	1	1	1	1	1	0	0	0
X	0	1	1	1	1	1	1	0	0	1
X	1	0	1	1	1	1	1	0	1	0
X	1	1	0	1	1	1	1	0	1	1
X	1	1	1	0	1	1	1	1	0	0
X	1	1	1	1	0	1	1	1	0	1
X	1	1	1	1	1	0	1	1	1	0
X	1	1	1	1	1	1	0	1	1	1

Fig. 9-13 Circuito lógico para um codificador octal para binário (8 para 3 linhas). Para operação correta, apenas uma entrada deve estar ativa de cada vez.

Codificadores de Prioridade

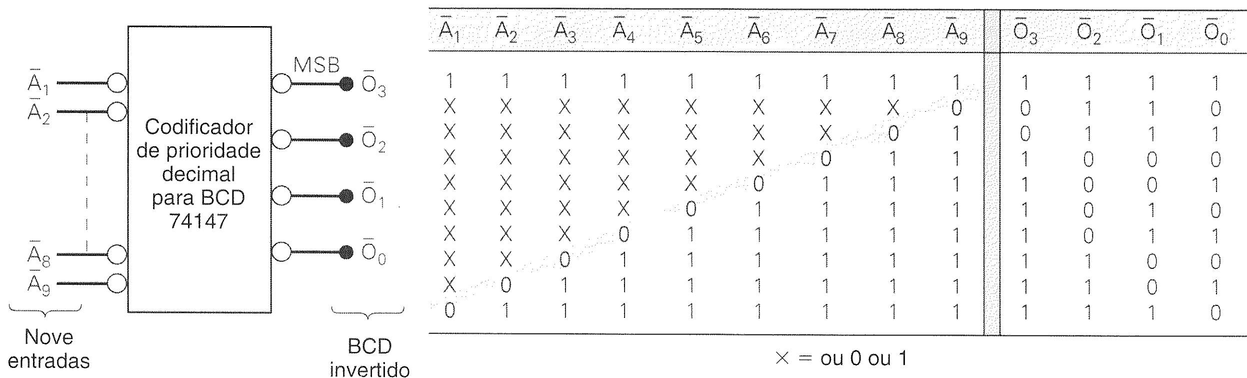
Este último exemplo identifica um inconveniente do circuito codificador simples da Fig. 9-13 quando mais do que uma entrada é ativada ao mesmo tempo. Uma versão modificada deste circuito, chamada de **codificador de prioridade**, inclui a lógica necessária para garantir que, quando duas ou mais entradas estiverem ativadas, o código de saída corresponderá à entrada com número mais alto. Por exemplo, quando tanto $\overline{A_3}$ quanto $\overline{A_5}$ estiverem em BAIXO, o código de saída será 101 (5). Analogamente, quando $\overline{A_6}$, $\overline{A_2}$ e $\overline{A_0}$ estiverem todas em BAIXO, o código de saída será 110 (6). O 74148, 74LS148 e o 74HC148 são codificadores de prioridade octal para binário.

Codificador de Prioridade Decimal para BCD 74147

A Fig. 9-14 mostra o símbolo lógico e a tabela-verdade para o 74147 (74LS147, 74HC147), que funciona como um codificador de prioridade decimal para BCD. Ele possui nove entradas ativas em BAIXO representando os dígitos decimais de 1 até 9, e produz o código BCD *invertido* correspondente à entrada ativada de número mais elevado.

Vamos examinar a tabela-verdade para ver como este CI funciona. A primeira linha na tabela mostra todas as entradas nos seus estados inativos em ALTO. Para esta condição, as saídas são 1111, que é o inverso de 0000, o código BCD para 0. A segunda linha na tabela indica um nível BAIXO em $\overline{A_9}$, que, independentemente dos estados das outras entradas, produzirá um código de saída 0110, que é o inverso de 1001, o código BCD para 9. A terceira linha mostra que um nível BAIXO em $\overline{A_8}$, desde que $\overline{A_9}$ esteja em ALTO, produzirá um código de saída 0111, o inverso de 1000, o código BCD para 8. Da mesma maneira, as linhas restantes na tabela mostram que um nível BAIXO em qualquer entrada, desde que todas as outras de mais alta ordem estejam em ALTO, produzirá o inverso do código BCD daquela entrada.

As saídas do 74147 normalmente estarão em ALTO quando nenhuma das entradas estiver sendo ativada. Isto cor-



$\overline{A_1}$	$\overline{A_2}$	$\overline{A_3}$	$\overline{A_4}$	$\overline{A_5}$	$\overline{A_6}$	$\overline{A_7}$	$\overline{A_8}$	$\overline{A_9}$	$\overline{O_3}$	$\overline{O_2}$	$\overline{O_1}$	$\overline{O_0}$
1	1	1	1	1	1	1	1	1	1	1	1	1
X	X	X	X	X	X	X	X	0	0	1	1	0
X	X	X	X	X	X	X	0	1	0	1	1	1
X	X	X	X	X	X	0	1	1	1	0	0	0
X	X	X	X	X	0	1	1	1	1	0	0	1
X	X	X	X	0	1	1	1	1	1	0	1	0
X	X	X	0	1	1	1	1	1	1	0	1	1
X	X	0	1	1	1	1	1	1	1	1	0	0
X	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0

× = ou 0 ou 1

Fig. 9-14 Codificador de prioridade decimal para BCD 74147.

responde à condição de entrada decimal 0. Não existe entrada $\overline{A_0}$, tendo em vista que o codificador assume o estado de entrada decimal 0 quando todas as outras entradas estão em ALTO. As saídas BCD invertidas do 74147 podem ser convertidas para BCD normal, colocando um INVERSOR para cada uma.

EXEMPLO 9-6

Determine os estados das saídas na Fig. 9-14 quando $\overline{A_5}$, $\overline{A_7}$ e $\overline{A_3}$ estiverem em BAIXO e todas as outras entradas estiverem em ALTO.

Solução

A tabela-verdade mostra que, quando $\overline{A_7}$ está BAIXO, os níveis em $\overline{A_5}$ e $\overline{A_3}$ não importam. Assim, as saídas serão 1000, respectivamente, o inverso de 0111 (7).

Codificador de Chaves

A Fig. 9-15 mostra como um 74147 pode ser usado como um *codificador de chaves*. As 10 chaves poderiam ser as teclas de um teclado de uma calculadora representando os dígitos de 0 a 9. As chaves são do tipo normalmente aberto, de modo que as entradas do codificador estão todas normalmente em ALTO e a saída BCD é 0000 (note os INVERSORes). Quando uma chave (tecla) de um dígito for pressionada, o circuito produzirá o código BCD desse dígito. Como o 74147 é um codificador de prioridade, acionamentos simultâneos de teclas produzirão o código BCD da tecla de maior número.

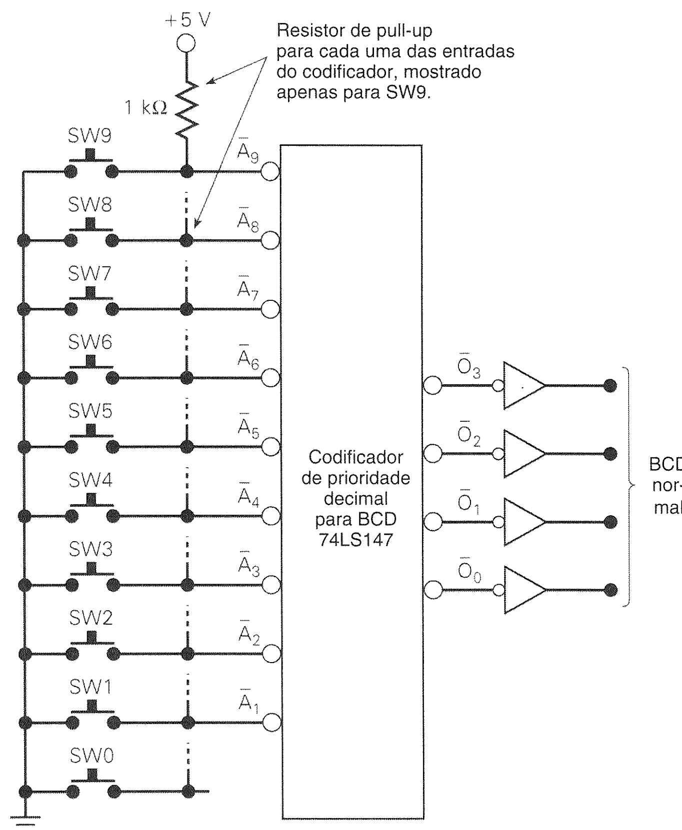


Fig. 9-15 Codificador de chaves decimal para BCD.

O codificador de chaves da Fig. 9-15 pode ser usado sempre que dados em BCD tiverem que ser fornecidos manualmente para um sistema digital. Um primeiro exemplo seria numa calculadora eletrônica, onde o operador pressionasse várias teclas em seguida para fornecer um número decimal. Em uma calculadora simples, o código BCD para cada dígito decimal seria colocado num registrador de armazenamento de quatro bits. Em outras palavras, quando a primeira tecla fosse pressionada, o código BCD para aquele dígito seria enviado para um registrador de quatro bits; quando a segunda tecla fosse pressionada, o código BCD daquele dígito seria enviado para *outro* registrador de quatro bits, e assim por diante. Deste modo, uma calculadora que pudesse operar com oito dígitos teria oito registradores de quatro bits para armazenar os códigos BCD para estes dígitos. Cada registrador de quatro bits acionaria um decodificador/driver e um display numérico, de modo que o número de oito dígitos pudesse ser mostrado.

A operação descrita anteriormente pode ser realizada com o circuito na Fig. 9-16. Este circuito recebe uma seqüência de três dígitos decimais do teclado, codifica-os em BCD e armazena-os em três registradores de saída. Os 12 flip-flops do tipo D, Q_0 a Q_{11}, são usados para receber e armazenar os códigos BCD para os dígitos. Q_8 a Q_{11} armazenam o código BCD para o dígito mais significativo (MSD), que é o primeiro digitado no teclado. Q_4 a Q_7 armazenam o segundo dígito, e Q_0 a Q_3 armazenam o terceiro. Os flip-flops *X*, *Y* e *Z* formam um contador em anel (Cap. 7) que controla a transferência de dados da saída do codificador para o registrador de saída apropriado. A porta OR produz uma saída em nível ALTO sempre que uma das teclas for pressionada. Esta saída pode ser afetada pela trepidação de contato da chave, que produziria vários pulsos antes de estabilizar no estado ALTO. O MONO é usado para neutralizar os efeitos da trepidação de contato da chave, sendo disparado na primeira transição positiva da saída da porta OR e permanecendo em ALTO por 20 ms, ultrapassando assim os efeitos da trepidação. A saída do MONO é o clock do contador em anel.

A operação do circuito é descrita a seguir, para o caso em que o número decimal 309 está sendo digitado:

1. A tecla LIMPA é pressionada. Isto limpa todos os FFs de armazenamento Q_0 a Q_{11} para 0. Isto também limpa os flip-flops *X* e *Y* e leva o flip-flop *Z* para 1, de modo que o contador em anel começa no estado 001.
2. A tecla LIMPA é liberada e a tecla "3" é pressionada. As saídas do codificador 1100 são invertidas para produzir 0011, o código BCD para 3. Estes valores binários são enviados para as entradas *D* dos três registradores de saída de quatro bits.
3. A saída OR vai para ALTO (pois duas de suas entradas estão em ALTO) e dispara a saída do MONO, $Q = 1$, por 20 ms. Após 20 ms, Q retorna para BAIXO e leva o contador em anel para o estado 100 (*X* vai para ALTO). A subida de *X* aciona as entradas *CLK* dos flip-flops Q_8 a Q_{11}, de modo que as saídas do codificador são

Fig. 9-16 Circuito para entrada pelo teclado de números de três dígitos em registradores de armazenamento.

transferidas para estes FFs. Isto é, $Q_{11} = 0$, $Q_{10} = 0$, $Q_9 = 1$ e $Q_8 = 1$. Note que os flip-flops Q_0 a Q_7 não são afetados, pois suas entradas de *CLK* não recebem uma transição positiva.

4. A tecla "3" é liberada e a saída da porta OR retorna para BAIXO. Então, a tecla "0" é pressionada. Isto produz o código BCD 0000, que é levado para a entrada dos três registradores.

5. A saída OR vai para ALTO em resposta à tecla "0" (note o INVERSOR) e dispara o MONO por 20 ms. Depois de 20 ms, o contador em anel avança para o estado 010 (*Y* vai para ALTO). A transição positiva em *Y* aciona as en-

tradas *CLK* dos flip-flops Q_4 a Q_7 e transfere o 0000 para estes FFs. Note que os flip-flops Q_0 a Q_3 e Q_8 a Q_{11} não são afetados pela transição em *Y*.

6. A tecla "0" é liberada e a saída da porta OR retorna para BAIXO. A tecla "9" é pressionada, produzindo as saídas BCD 1001, que são levadas para os registradores de armazenamento.
7. A saída OR vai para ALTO novamente, disparando o MONO, que por sua vez aciona o contador em anel para o estado 001 (*Z* vai para ALTO). A transição positiva em *Z* aciona as entradas *CLK* de Q_0 a Q_3 e transfere 1001 para estes FFs. Os outros FFs de armazenamento não são afetados.
8. Neste ponto, o registrador de armazenamento contém 001100001001, começando em Q_{11}. Este é o código BCD de 309. Estes registradores de saída são ligados a decodificadores/drivers que acionam os displays apropriados para indicar os dígitos decimais 309.
9. As saídas dos FFs de armazenamento também acionam outros circuitos no sistema. Em uma calculadora, por exemplo, estas saídas seriam enviadas para a seção aritmética para serem processadas.

Diversos problemas no final do capítulo tratarão de alguns outros aspectos deste circuito, incluindo exercícios de depuração.

Questões de Revisão

1. Como um codificador difere de um decodificador?
2. Em que um codificador de prioridade difere de um codificador comum?
3. Quais serão as saídas na Fig. 9-15 quando SW6, SW5 e SW2 estiverem fechadas?
4. Descreva as funções de cada uma das seguintes partes do circuito para entrada de dados pelo teclado da Fig. 9-16.
   (a) Porta OR
   (b) Codificador 74147
   (c) Monoestável
   (d) Flip-flops *X*, *Y*, *Z*
   (e) Flip-flops Q_0 a Q_{11}

9-5 SÍMBOLOS IEEE/ANSI

Vamos analisar agora os símbolos IEEE/ANSI para diversos decodificadores e codificadores. A Fig. 9-17(a) mostra o símbolo IEEE/ANSI para o decodificador 7442. Ele é bastante direto. Note a identificação BCD/DEC, que denota que ele é um decodificador BCD para decimal. Repare também na maneira pela qual as entradas e saídas são numeradas dentro do bloco. *Não* são números dos pinos do CI. Por exemplo, as entradas do 7442 têm os números 1, 2, 4 e 8, respectivamente, dentro do retângulo. Eles indicam os pesos relativos de cada entrada no número BCD.

O símbolo IEEE/ANSI para o 7445, na Fig. 9-17(b), é similar ao do 7442, com duas diferenças. A primeira é que cada saída tem um símbolo de diamante sublinhado para indicar sua estrutura em coletor aberto. A segunda é o triângulo no meio do símbolo, que indica que este dispositivo é um buffer ou um driver com capacidade de tensão e/ou corrente maior do que o normal.

A Fig. 9-17(c) é o símbolo para o decodificador 74LS138. Note a identificação BIN/OCT para indicar sua função de decodificação de binário para octal. Note também como as três entradas de habilitação são combinadas em um bloco AND para produzir um sinal de habilitação geral, *EN*.

A Fig. 9-18 mostra o símbolo IEEE/ANSI para o CI codificador 74147. O rótulo HPRI/BCD indica que a função deste CI é converter a entrada ativa de mais alta prioridade (do inglês, *highest priority*) para seu código BCD. Novamente, repare como as entradas e saídas são numeradas dentro do bloco.

Questões de Revisão

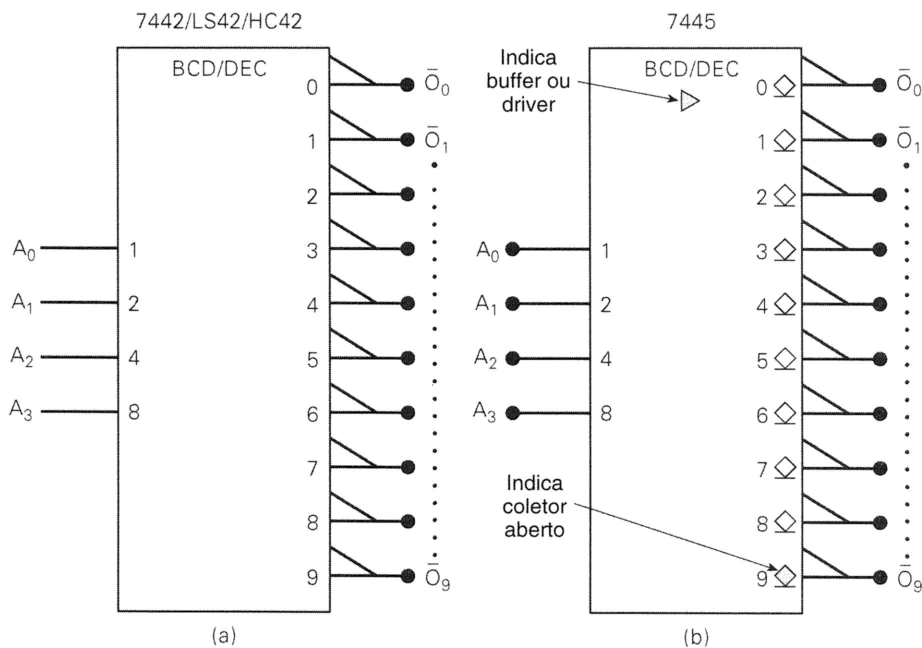
1. O que significa o rótulo BCD/DEC?
2. O que significa o rótulo HPRI?
3. O que significa o símbolo ▷ dentro de um bloco?

9-6 PESQUISA DE FALHAS

À medida que os circuitos e sistemas se tornam cada vez mais complexos, o número de possíveis causas de falhas, obviamente, aumenta. Embora o procedimento para isolar e corrigir falhas permaneça essencialmente o mesmo, a aplicação do processo de **observação/análise** é mais importante para circuitos complexos, porque ajuda o pesquisador responsável pela depuração a limitar a localização da falha a uma pequena área do circuito. Isto reduz para uma quantidade razoável os testes e os resultados que devem ser analisados. Entendendo a operação do circuito, observando os sintomas da falha e raciocinando sobre a operação, o pesquisador freqüentemente pode prever as falhas possíveis mesmo antes de usar uma ponta de prova lógica ou um osciloscópio. Este processo de observação/análise faz com que os pesquisadores mais inexperientes fiquem hesitantes em aplicá-lo, provavelmente devido à grande variedade e capacidade dos modernos equipamentos de teste disponíveis para eles. É fácil tornar-se confiante em demasia com estes instrumentos e não usar adequadamente as habilidades analíticas e o raciocínio humanos.

Os exemplos a seguir ilustram como o processo de observação/análise pode ser aplicado. Muitos problemas de depuração, ao final do capítulo, fornecerão ao estudante a oportunidade de desenvolver suas habilidades para aplicar este processo.

Uma outra estratégia vital de pesquisa de falhas é conhecida como **dividir e conquistar**. Ela é usada para identificar a localização do problema após a observação/análise ter gerado um certo número de possibilidades. Um método menos eficiente seria investigar cada causa possível, uma por vez. O método de dividir e conquistar determina um ponto do circuito que pode ser testado, dividindo portanto o número total de causas possíveis pela metade. Em sistemas simples isto pode parecer desnecessário, mas, conforme a complexidade aumenta, o número total de causas possíveis também aumenta. Se existirem oito causas possí-



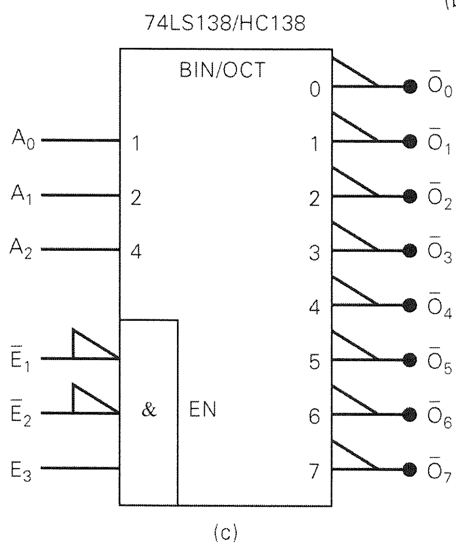


Fig. 9-17 Símbolos IEEE/ANSI para diversos decodificadores.

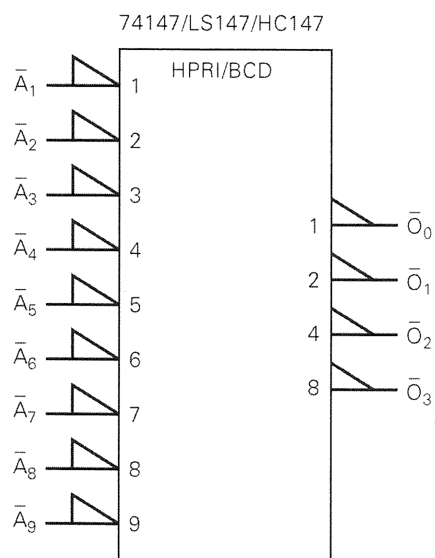


Fig. 9-18 Símbolo IEEE/ANSI para o codificador 74147.

veis, então deveria ser feito um teste que eliminasse quatro delas. O próximo teste deveria eliminar mais duas, e o terceiro teste deveria identificar o problema.

EXEMPLO 9-7

Uma estudante testa o circuito da Fig. 9-4 utilizando um conjunto de chaves para aplicar o código de entrada, A_4 a A_0. Ela gera cada um dos possíveis códigos de entrada e verifica a saída correspondente do decodificador, para constatar se ela está sendo ativada. Observa que todas as saídas ímpares respondem corretamente, mas todas as saídas pares falham em responder quando seus códigos são aplicados. Quais são as falhas mais prováveis?

Solução

Numa situação em que tantas saídas estão falhando, não é razoável supor que cada uma delas tenha uma falha. É mais provável que alguma condição de falha na entrada esteja causando as falhas nas saídas. O que todas as saídas pares têm em comum? Os códigos de entrada para várias delas estão listados na Tabela 9-1

TABELA 9-1

Saída	Código de Entrada
$\overline{O_0}$	00000
$\overline{O_4}$	00100
$\overline{O_{14}}$	01110
$\overline{O_{18}}$	10010

Evidentemente, cada saída par necessita de um código de entrada com $A_0 = 0$, de modo a ser ativada. Logo, as falhas mais prováveis são aquelas que impedem A_0 de ir para BAIXO. Elas incluem:

1. Uma chave ruim conectada na entrada A_0
2. Uma interrupção na ligação entre a chave e a linha A_0
3. Um curto circuito externo da linha A_0 com V_{CC}
4. Um curto circuito interno para V_{CC} nas entradas A_0 de qualquer dos chips decodificadores

Através da observação e da análise, a estudante identificou diversas causas possíveis. As causas #1 e #2 estão relacionadas à geração do endereço. As causas #3 e #4 estão no circuito de decodificação. O circuito pode ser dividido abrindo-se a conexão entre a chave menos significativa e a entrada A_0, como mostrado na Fig. 9-19. Uma ponta de prova lógica pode ser usada para ver se a chave pode gerar um nível BAIXO assim como um nível ALTO. Independentemente do resultado, duas das quatro causas possíveis foram eliminadas.

Deste modo, a falha está restrita a uma área específica do circuito. A falha exata pode ser rastreada com as técnicas de teste e medição com as quais já estamos familiarizados.

EXEMPLO 9-8

Um estudante liga as saídas de um contador BCD nas entradas do decodificador/driver da Fig. 9-8. Ele aplica pulsos ao contador a uma taxa bastante lenta e observa o display a LEDs, que está mostrado a seguir, à medida que o contador avança desde 0000 até 1001. Examine cuidadosamente a seqüência observada e tente prever a falha mais provável.

CONTAGEM	0	1	2	3	4	5	6	7	8	9
Display observado	0	1	[illegible]	3	[illegible]	[illegible]	6	7	8	[illegible]
Display esperado	0	1	2	3	4	5	6	7	8	9

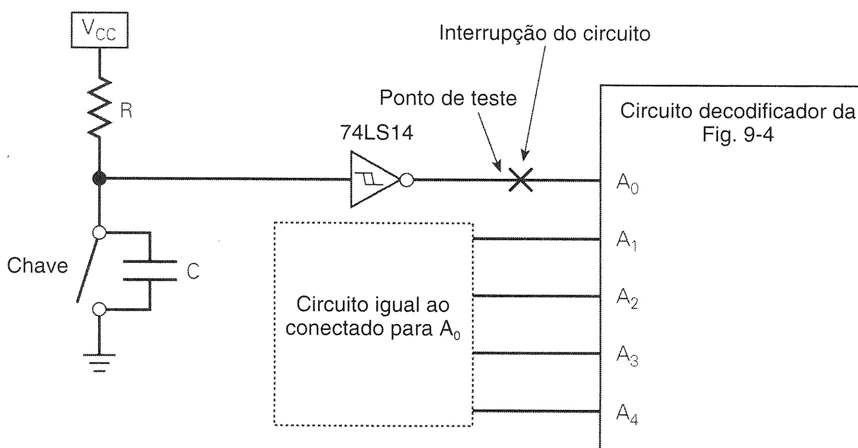


Fig. 9-19 Circuito para depuração do Exemplo 9-7.

Solução

Comparando o display observado com o esperado para cada contagem, constatamos diversos pontos importantes:

- Para aquelas contagens em que o display observado está incorreto, o display observado não é um dos padrões de segmento que corresponde a contagens maiores do que 1001.
- Isto exclui um contador defeituoso ou erros de ligação do contador para o decodificador/driver.
- Os padrões de segmento corretos (0, 1, 3, 6, 7 e 8) têm em comum que os segmentos *e* e *f* estão ambos ligados ou desligados.
- Os padrões de segmento incorretos têm em comum que os segmentos *e* e *f* estão em estados opostos, e, se trocarmos o estado destes dois segmentos, o padrão correto é obtido.

Considerando estes pontos, somos levados a concluir que provavelmente o estudante "trocou" as conexões dos segmentos *e* e *f*.

9-7 MULTIPLEXADORES (SELETORES DE DADOS)

Um moderno sistema de som estéreo pode ter uma chave que seleciona música de uma das quatro fontes: fita cassete, compact disc (CD), toca-disco ou um rádio. A chave seleciona um dos sinais eletrônicos de uma destas quatro fontes e o envia para o amplificador de potência e os alto-falantes. De um modo simplificado, isto é o que um **multiplexador (MUX)** faz: ele seleciona um entre vários sinais de entrada e o envia para a saída.

Um *multiplexador digital* ou *seletor de dados* é um circuito lógico que aceita diversos dados digitais de entrada e seleciona um deles, em um certo instante, para a saída. O roteamento do sinal de entrada desejado para a saída é controlado pelas entradas de SELEÇÃO (freqüentemente chamadas de entradas de ENDEREÇO). A Fig. 9-20 mostra o diagrama funcional de um multiplexador digital geral. As

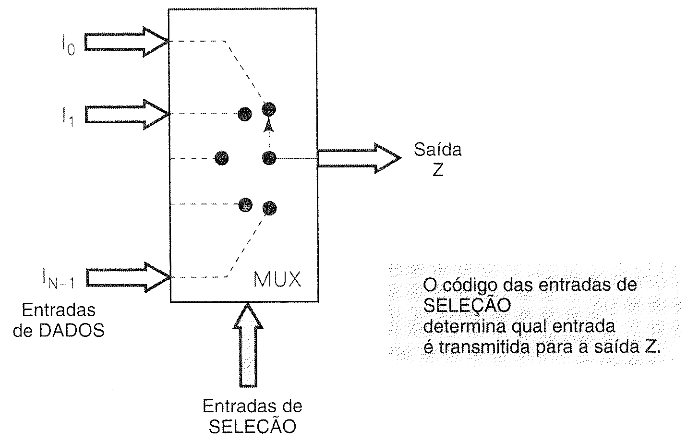


Fig. 9-20 Diagrama funcional de um multiplexador (MUX) digital.

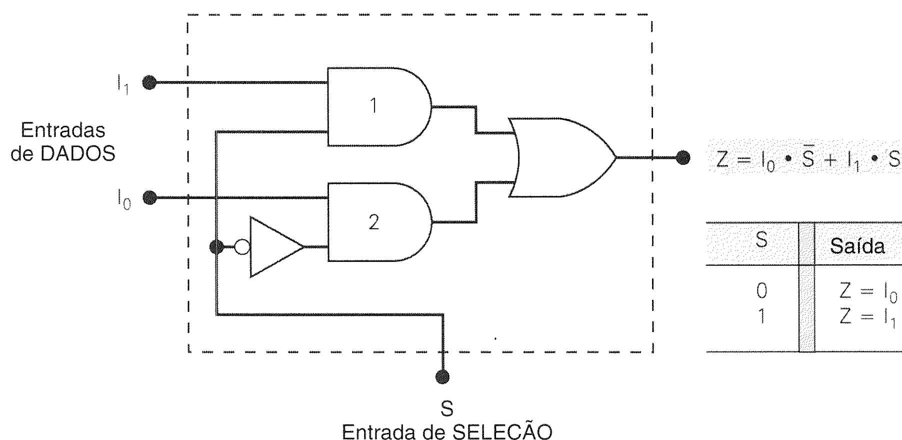
entradas e saídas são desenhadas como setas mais largas em vez de linhas; isto indica que elas podem, na verdade, representar mais do que uma linha de sinal.

O multiplexador atua como uma chave digital controlada de várias posições, onde o código digital aplicado nas entradas de SELEÇÃO controla qual entrada de dados será chaveada para a saída. Por exemplo, a saída Z será igual à entrada de dados I_0 para um determinado código de SELEÇÃO; Z será igual a I_1 para um outro determinado código de SELEÇÃO; e assim por diante. Em outras palavras, um multiplexador seleciona 1 entre N dados de entrada e transmite o dado selecionado para um único canal de saída. Isto é chamado de **multiplexação.**

Multiplexador Básico de Duas Entradas

A Fig. 9-21 mostra o circuito lógico para um multiplexador de duas entradas, com as entradas de dados I_0 e I_1, e uma entrada de SELEÇÃO S. O nível lógico aplicado na entrada S determina qual das portas AND é habilitada, de modo que seu dado passe pela porta OR para a saída Z. Analisando de outra maneira, a expressão booleana para a saída é

$$Z = I_0\overline{S} + I_1S$$



S	Saída
0	$Z = I_0$
1	$Z = I_1$

Fig. 9-21 Multiplexador de duas entradas.

Com $S = 0$, esta expressão se torna

$$Z = I_0 \cdot 1 + I_1 \cdot 0 \qquad \text{\{porta 2 habilitada\}}$$
$$= I_0$$

que indica que Z será idêntica ao sinal de entrada I_0, que pode ser um nível lógico fixo ou um sinal lógico variante no tempo. Com $S = 1$, a expressão se torna

$$Z = I_0 \cdot 0 + I_1 \cdot 1 = I_1 \qquad \text{\{porta 1 habilitada\}}$$

mostrando que a saída Z será idêntica ao sinal de entrada I_1.

Um exemplo de onde um MUX de duas entradas pode ser usado é um computador que utiliza dois sinais diferentes de CLOCK: um clock de alta velocidade (digamos, 10 MHz) para alguns programas, e um clock lento (digamos, 4,77 MHz) para outros. Utilizando o circuito da Fig. 9-21, o clock de 10 MHz seria ligado em I_0, e o clock de 4,77 MHz, em I_1. Um sinal da área de controle lógico do computador acionaria a entrada de SELEÇÃO, para controlar qual dos sinais de clock apareceria na saída Z, para ser levado a outros circuitos do computador.

Multiplexador de Quatro Entradas

A mesma idéia básica pode ser usada para formar o multiplexador de quatro entradas mostrado na Fig. 9-22. Neste caso, existem quatro entradas, as quais são seletivamente transmitidas para a saída, de acordo com as quatro combinações possíveis das entradas de seleção S_1S_0. Cada entrada de dados é selecionada com uma combinação diferente de níveis das entradas de seleção. I_0 é selecionado com $\overline{S_1}\,\overline{S_0}$, de modo que I_0 passará pela sua porta AND para a saída Z somente quando $S_1 = 0$ e $S_0 = 0$. A tabela na figura fornece as saídas para os outros três códigos de seleção.

Multiplexadores de duas, quatro, oito e dezesseis entradas estão disponíveis nas famílias lógicas TTL e CMOS. Estes CIs básicos podem ser combinados para a multiplexação de um maior número de entradas.

Multiplexador de Oito Entradas

A Fig. 9-23(a) mostra o diagrama lógico para o multiplexador de oito entradas 74151 (74LS151, 74HC151). Este multiplexador tem uma entrada de habilitação $\overline{E}$ e fornece tanto a saída normal quanto a saída invertida. Quando $\overline{E} = 0$, as entradas de seleção $S_2S_1S_0$ selecionarão uma das entradas de dados (de I_0 a I_7) para passar para a saída Z. Quando $\overline{E} = 1$, o multiplexador está inibido, de modo que $Z = 0$, não importando o código de seleção de entrada. Esta operação está resumida na Fig. 9-23(b), e o símbolo lógico do 74151 é mostrado na Fig. 9-23(c).

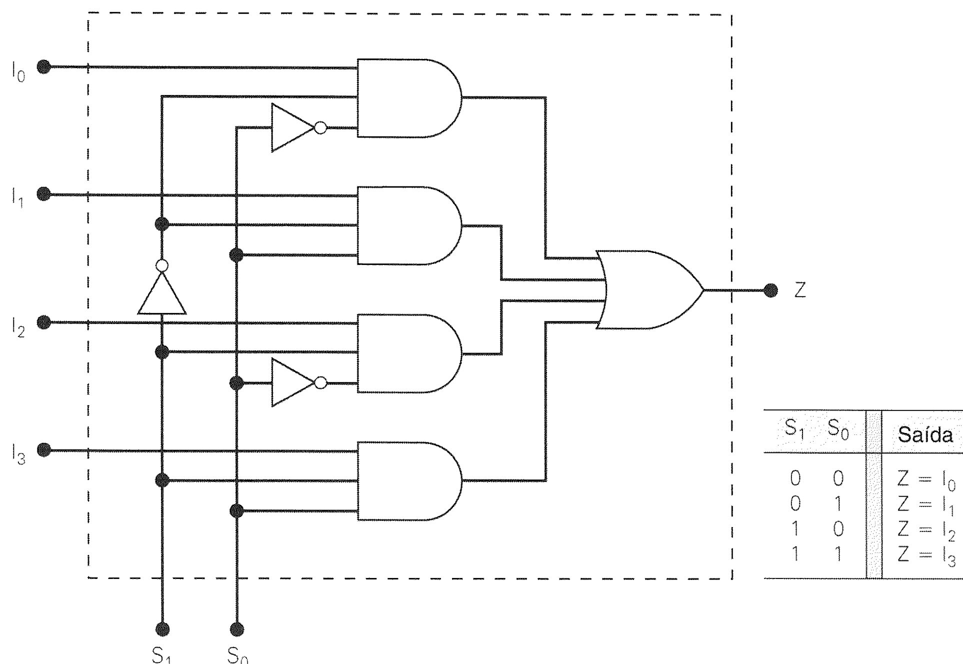
EXEMPLO 9-9

O circuito na Fig. 9-24 utiliza dois 74HC151s, um INVERSOR e uma porta OR. Descreva a operação do circuito.

Solução

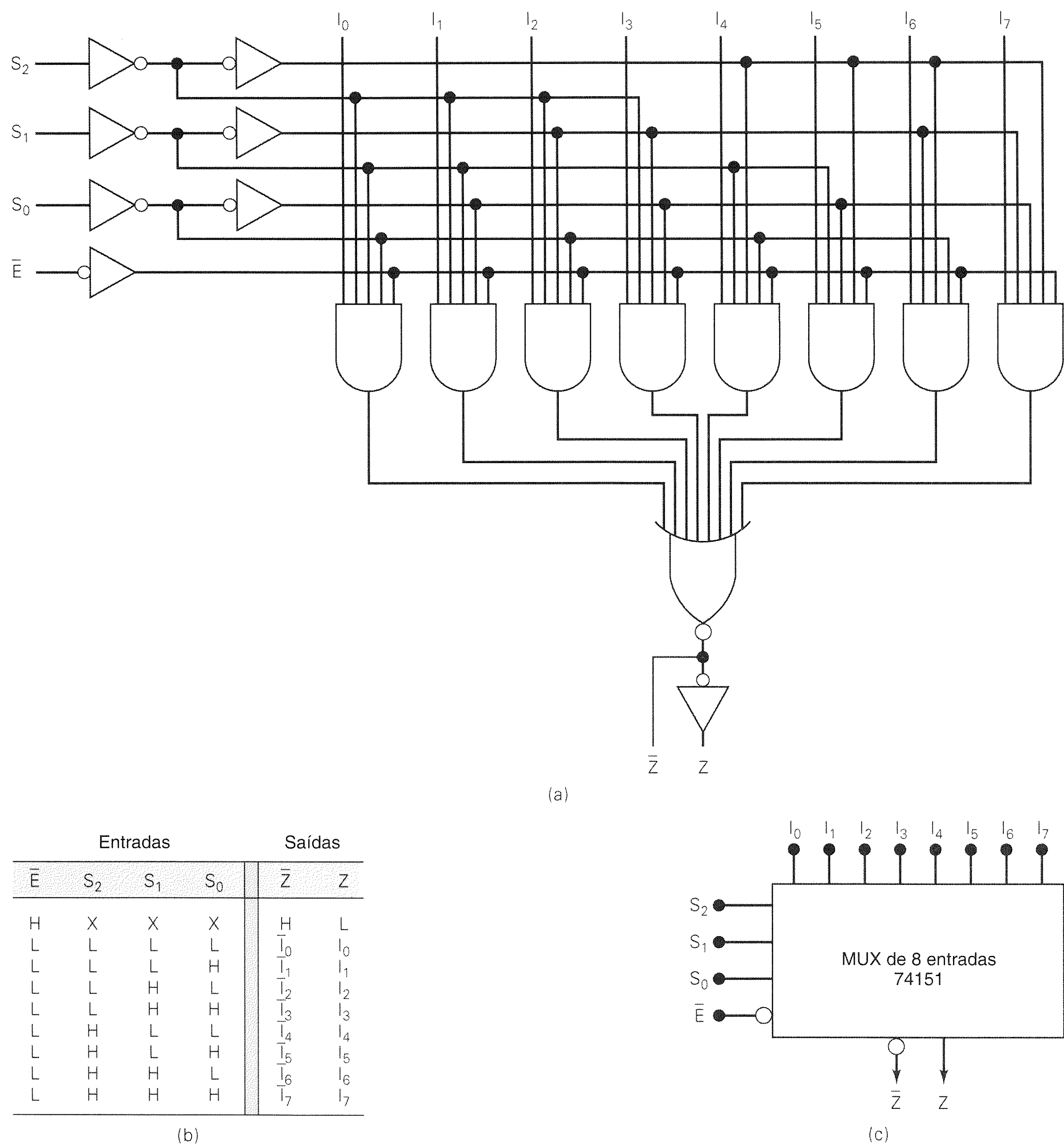
Este circuito tem, no total, 16 entradas de dados, oito aplicadas em cada multiplexador. As saídas dos dois multiplexadores são combinadas na porta OR para produzir uma única saída X. O circuito funciona como um multiplexador de 16 entradas. As quatro entradas de seleção $S_3S_2S_1S_0$ selecionam uma entre as 16 entradas para passá-la para X.

A entrada S_3 determina qual multiplexador é habilitado. Quando $S_3 = 0$, o multiplexador superior é habilitado e as entradas $S_2S_1S_0$ determinam qual das suas entradas de dados aparecerá na sua saída e será passada pela porta OR para X. Quando $S_3 = 1$, o multiplexador inferior é habilita-



S_1	S_0	Saída
0	0	$Z = I_0$
0	1	$Z = I_1$
1	0	$Z = I_2$
1	1	$Z = I_3$

Fig. 9-22 Multiplexador de quatro entradas.



Entradas				Saídas	
$\bar{E}$	S_2	S_1	S_0	$\bar{Z}$	Z
H	X	X	X	H	L
L	L	L	L	$\bar{I}_0$	I_0
L	L	L	H	$\bar{I}_1$	I_1
L	L	H	L	$\bar{I}_2$	I_2
L	L	H	H	$\bar{I}_3$	I_3
L	H	L	L	$\bar{I}_4$	I_4
L	H	L	H	$\bar{I}_5$	I_5
L	H	H	L	$\bar{I}_6$	I_6
L	H	H	H	$\bar{I}_7$	I_7

(b)

Fig. 9-23 (a) Diagrama lógico para o multiplexador 74151; (b) tabela-verdade; (c) símbolo lógico. (Cortesia da Fairchild, uma empresa da Schlumberger.)

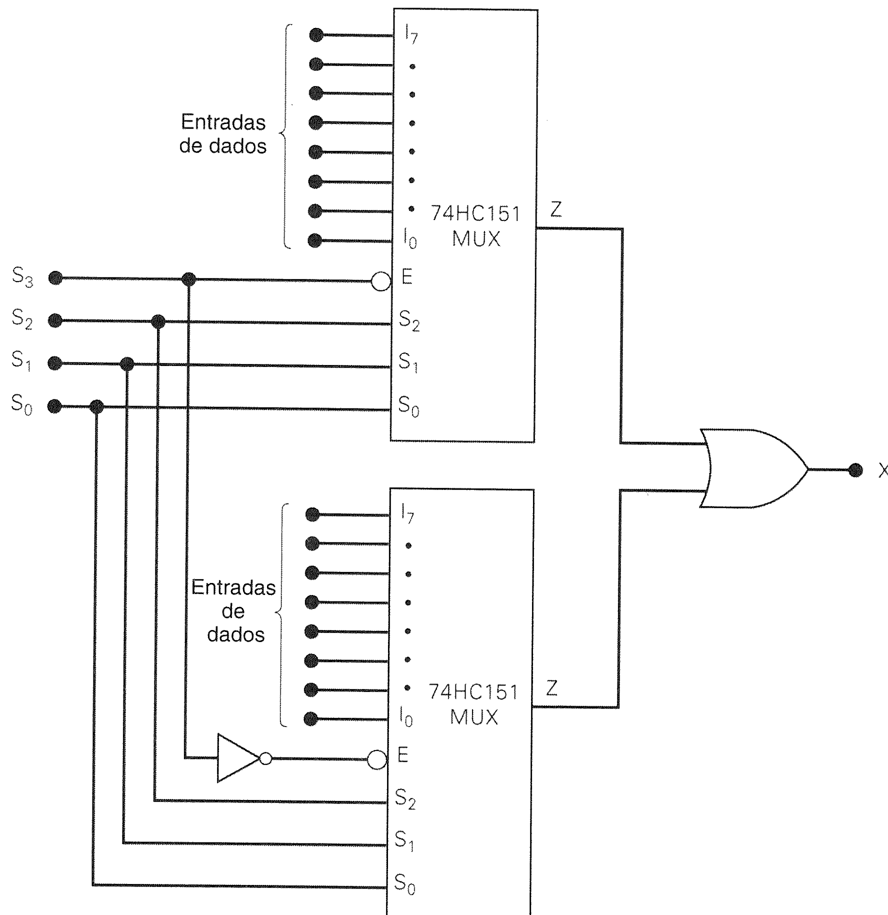


Fig. 9-24 Exemplo 9-9: dois 74HC151s combinados para formar um multiplexador de 16 entradas.

do e as entradas $S_2S_1S_0$ selecionam uma das suas entradas de dados para passar para a saída X.

MUX Quádruplo de Duas Entradas (74157/LS157/HC157)

O 74157 é um CI multiplexador muito útil, que contém quatro multiplexadores de duas entradas, como aquele que aparece na Fig. 9-21. O diagrama lógico para o 74157 é mostrado na Fig. 9-25(a). Note a maneira pela qual as entradas de dados e as saídas são rotuladas.

EXEMPLO 9-10

Determine as condições de entrada necessárias para cada saída Z assumir o nível lógico da sua correspondente entrada I_0. Repita para I_1.

Solução

Em primeiro lugar, a entrada de habilitação deve estar ativa; isto é, $\overline{E} = 0$. Para que Z_a seja igual a I_{0a}, a entrada de seleção deve estar em BAIXO. Estas mesmas condições produzem $Z_b = I_{0b}$, $Z_c = I_{0c}$ e $Z_d = I_{0d}$.

Com $\overline{E} = 0$ e $S = 1$, as saídas Z seguem o conjunto de entradas I_1; isto é, $Z_a = I_{1a}$, $Z_b = I_{1b}$, $Z_c = I_{1c}$ e $Z_d = I_{1d}$.

Todas as saídas estarão inibidas (BAIXO) quando $\overline{E} = 1$.

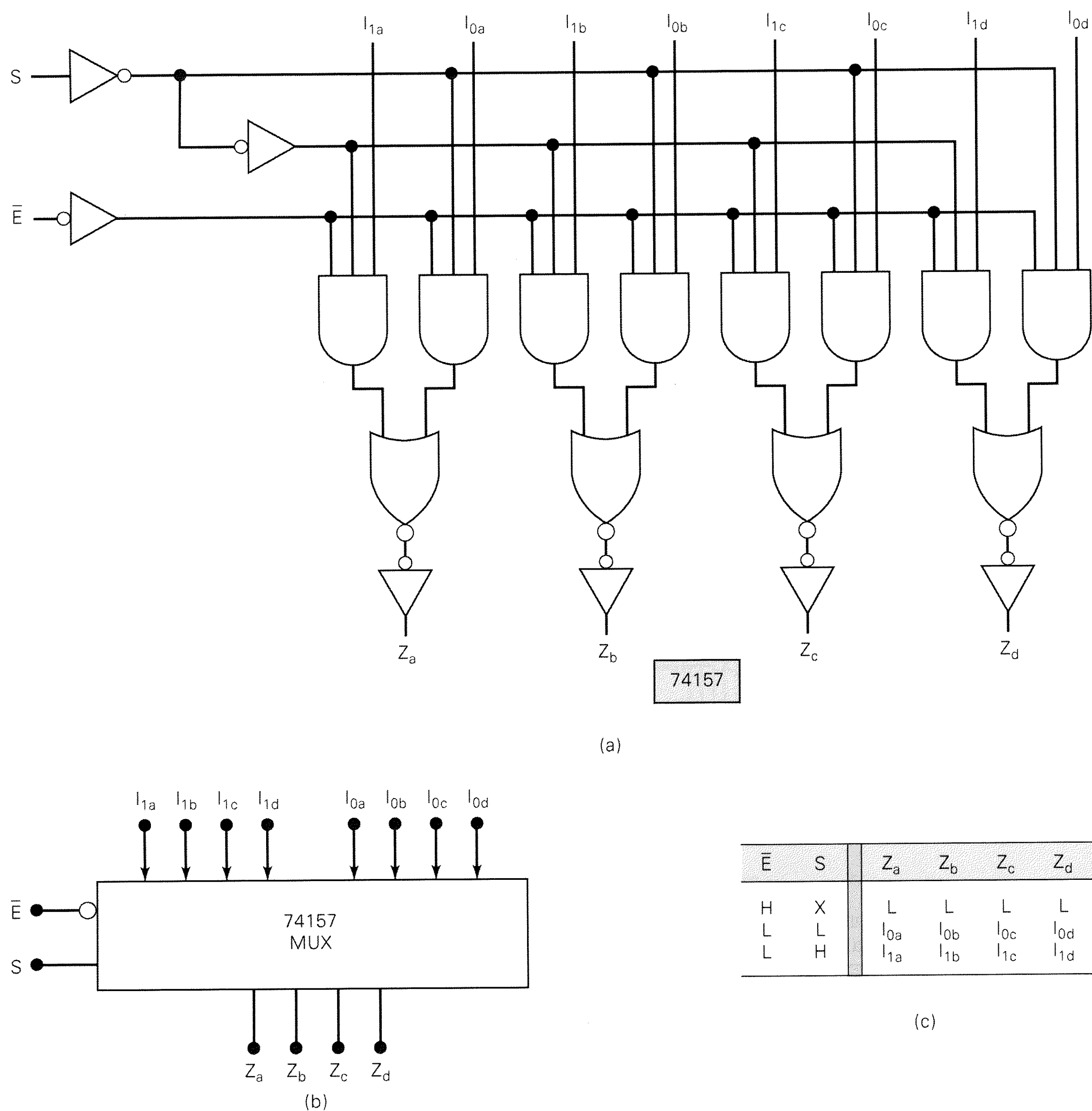
É conveniente pensar neste multiplexador como sendo um simples multiplexador de duas entradas, mas com cada entrada tendo quatro linhas e a saída com quatro linhas. As quatro linhas de saída apresentam um dos dois conjuntos de quatro linhas de entrada, sob o controle da entrada de seleção. Esta operação está representada pelo símbolo lógico do 74157 na Fig. 9-25(b).

Questões de Revisão

1. Qual é a função das entradas de seleção de um multiplexador?
2. Um certo multiplexador pode chavear um dentre 32 dados de entrada para sua saída. Quantas entradas diferentes tem este MUX?

9-8 APLICAÇÕES DE MULTIPLEXADORES

Circuitos multiplexadores encontram numerosas e diversas aplicações em sistemas digitais de todos os tipos. Estas aplica-



Ē	S	Z_a	Z_b	Z_c	Z_d
H	X	L	L	L	L
L	L	I_{0a}	I_{0b}	I_{0c}	I_{0d}
L	H	I_{1a}	I_{1b}	I_{1c}	I_{1d}

Fig. 9-25 (a) Diagrama lógico para o multiplexador 74157; (b) símbolo lógico; (c) tabela-verdade. (Cortesia da Fairchild, uma empresa da Schlumberger.)

ções incluem seleção de dados, roteamento de dados, seqüenciamento de operações, conversões paralelo-série, geração de formas de onda e de funções lógicas. Vamos estudar algumas destas aplicações nesta seção e muitas outras nos problemas ao final do capítulo.

Roteamento de Dados

Multiplexadores podem rotear dados de diversas fontes para um destino. Uma aplicação típica utiliza multiplexadores 74LS157 para selecionar e mostrar o conteúdo de um entre dois contadores BCD usando apenas um *único* conjunto de decodificador/driver e display a LEDs. A organização do circuito é mostrada na Fig. 9-26.

Cada contador consiste em dois estágios BCD em cascata, e cada um é acionado por seu próprio sinal de clock. Quando a linha SELECIONA_CONTADOR estiver em ALTO, as saídas do contador 1 passarão pelos multiplexadores para os decodificadores/drivers para serem mostradas nos displays a LEDs. Quando SELECIONA_CONTADOR = 0, as saídas do contador 2 passarão pelos multiplexadores para os displays. Deste modo, o conteúdo decimal de um ou de outro contador será apresentado sob o controle da entrada SELECIONA_CONTADOR. Uma situação comum onde isto poderia ser usado é um relógio digital. Os circuitos do reló-

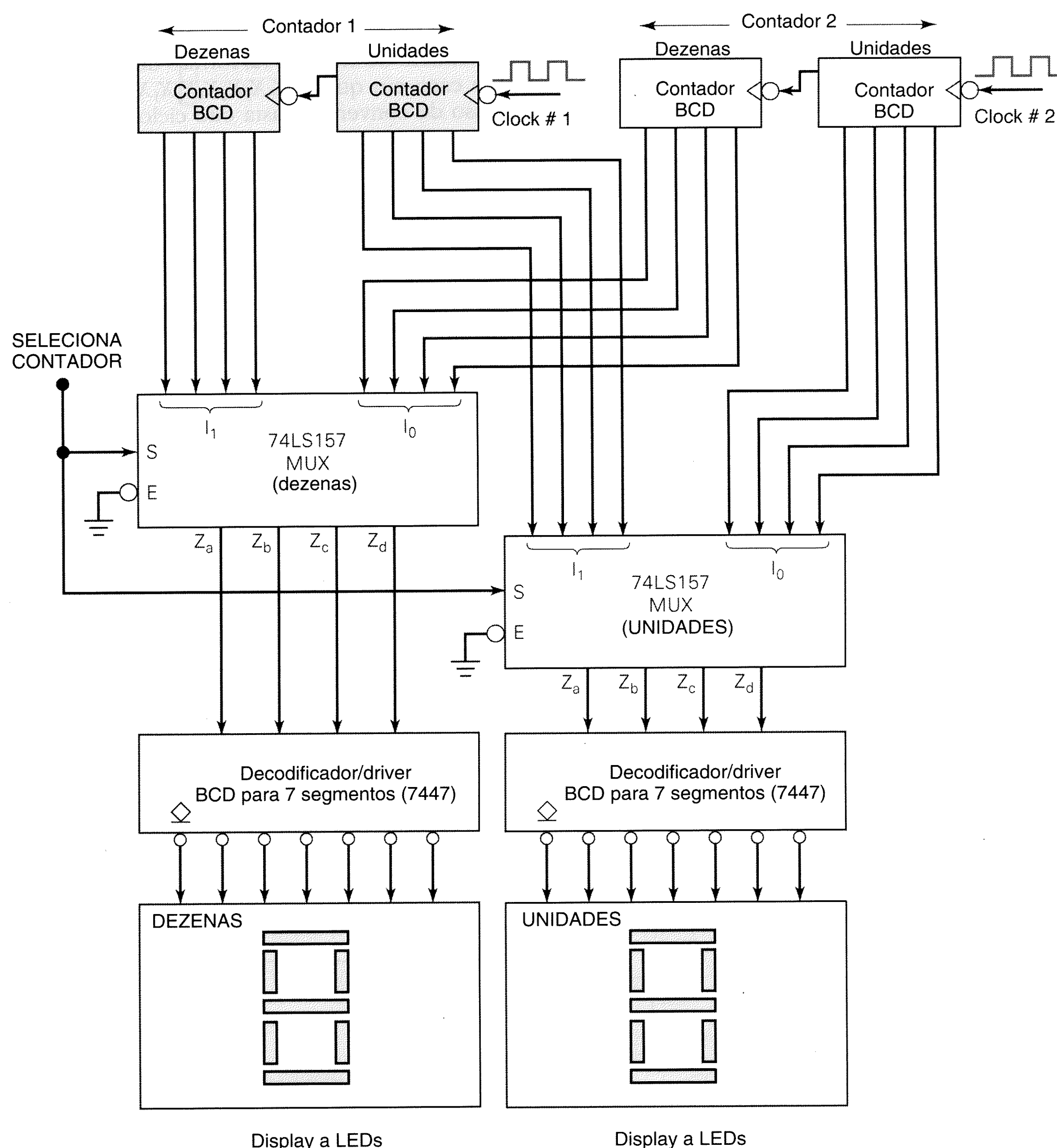


Fig. 9-26 Sistema para mostrar o valor de dois contadores BCD, um de cada vez.

gio digital contêm muitos contadores e registradores que tratam dos segundos, minutos, horas, dias, meses, alarme, e assim por diante. Um esquema de multiplexação, tal como este, permite que diferentes dados sejam apresentados no número limitado de displays.

O propósito da técnica de multiplexação, da maneira como é usada aqui, é *compartilhar* os circuitos decodificadores/drivers e displays entre os dois contadores, em vez de ter um conjunto de decodificadores/drivers e displays para cada contador. Isto acarreta uma diminuição do número de conexões a serem realizadas, especialmente quando mais estágios BCD forem adicionados para cada contador. Ainda mais importante, isto representa uma diminuição significativa de consumo, pois decodificadores/drivers e displays a LEDs absorvem, relativamente, grandes quantidades de corrente da fonte V_{CC}. Naturalmente, esta técnica tem a limitação de que apenas o conteúdo de um contador pode ser mostrado por vez. Entretanto, em várias aplicações isto não é inconveniente. Uma chave mecânica poderia ter sido usada para realizar a função de chavear primeiro um contador e depois o outro para os decodificadores/drivers e displays, mas o número de contatos necessários, a complexidade das ligações e as dimensões físicas seriam desvantagens em relação ao método puramente lógico da Fig. 9-26.

Conversão Paralelo-Série

Muitos sistemas digitais processam os dados binários na forma paralela (todos os bits simultaneamente) porque é

mais rápido. Entretanto, quando os dados devem ser transmitidos por distâncias relativamente longas, o esquema paralelo não é desejável, pois necessita de um grande número de linhas de transmissão. Por esta razão, dados binários ou informações na forma paralela são freqüentemente convertidos para o formato serial antes de serem transmitidos para um destino remoto. Um método para realizar esta **conversão paralelo-série** utiliza um multiplexador, conforme ilustrado na Fig. 9-27.

Os dados são apresentados na forma paralela nas saídas do registrador *X* e colocados no multiplexador de oito entradas. Um contador de três bits (módulo 8) é usado para fornecer os bits do código de seleção $S_2S_1S_0$, de modo que ele cicla desde de 000 até 111, conforme os pulsos de clock vão sendo aplicados. Desta maneira, a saída do multiplexador será X_0 durante o primeiro período de clock, X_1 durante o segundo período de clock, e assim por diante. A saída *Z* tem uma forma de onda que é a representação serial do dado paralelo de entrada. As formas de onda da figura são para o caso em que $X_7X_6X_5X_4X_3X_2X_1X_0$ = 10110101. Este processo de conversão gasta oito ciclos de clock no total. Note que X_0 (o LSB) é transmitido primeiro e X_7 (MSB) é transmitido por último.

Seqüenciamento de Operações

O circuito da Fig. 9-28 utiliza um multiplexador de oito entradas como parte de um seqüenciador de controle de sete passos, onde cada passo atua numa porção do processo físico que está sendo controlado. Isto poderia ser, por exemplo, um processo que misturasse dois ingredientes líquidos e então cozinhasse a mistura. O circuito também usa um decodificador de 3 para 8 linhas e um contador binário de módulo 8. A operação é descrita a seguir:

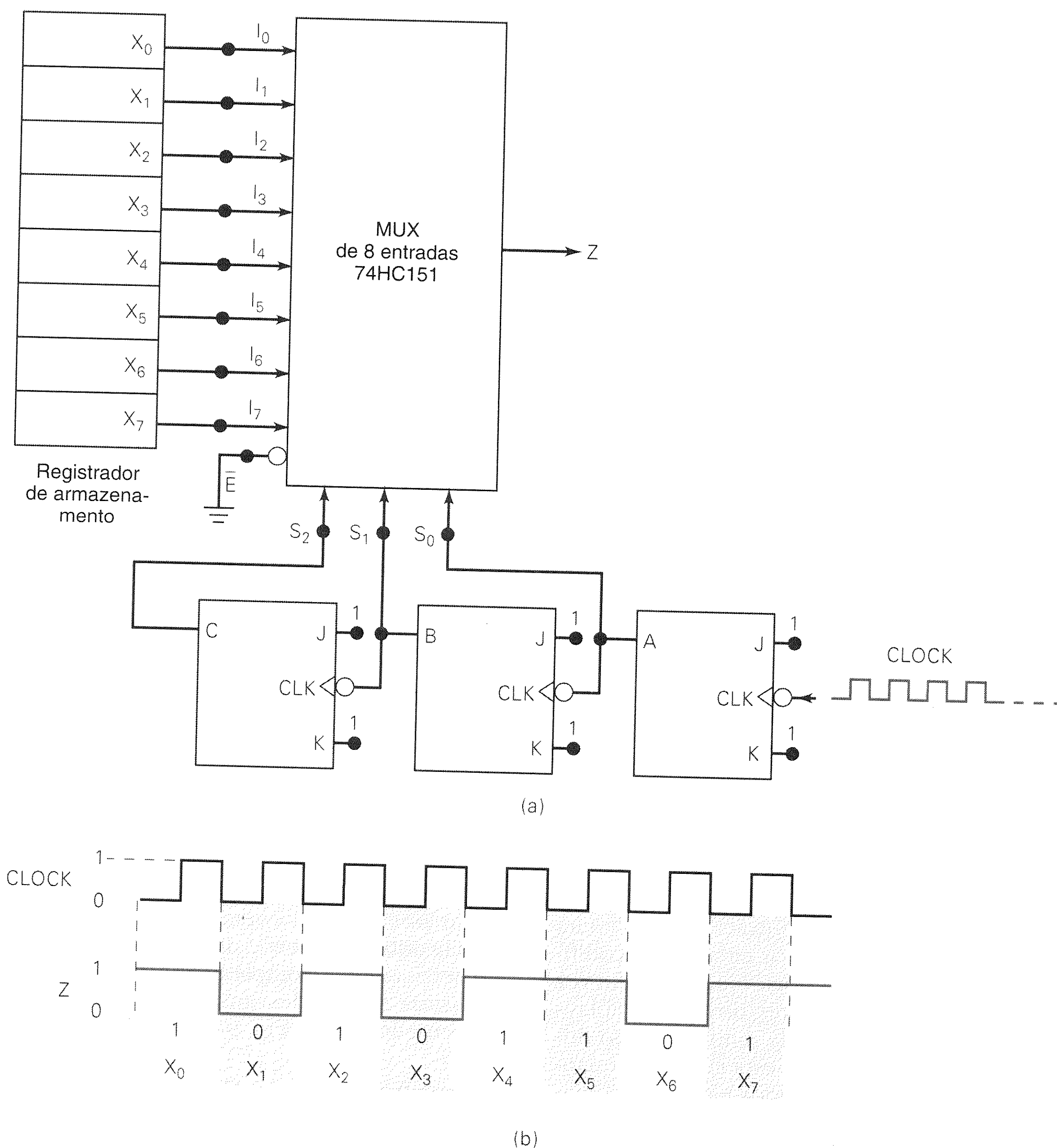


Fig. 9-27 (a) Conversor paralelo-série; (b) formas de onda para $X_7X_6X_5X_4X_3X_2X_1X_0$ = 10110101.

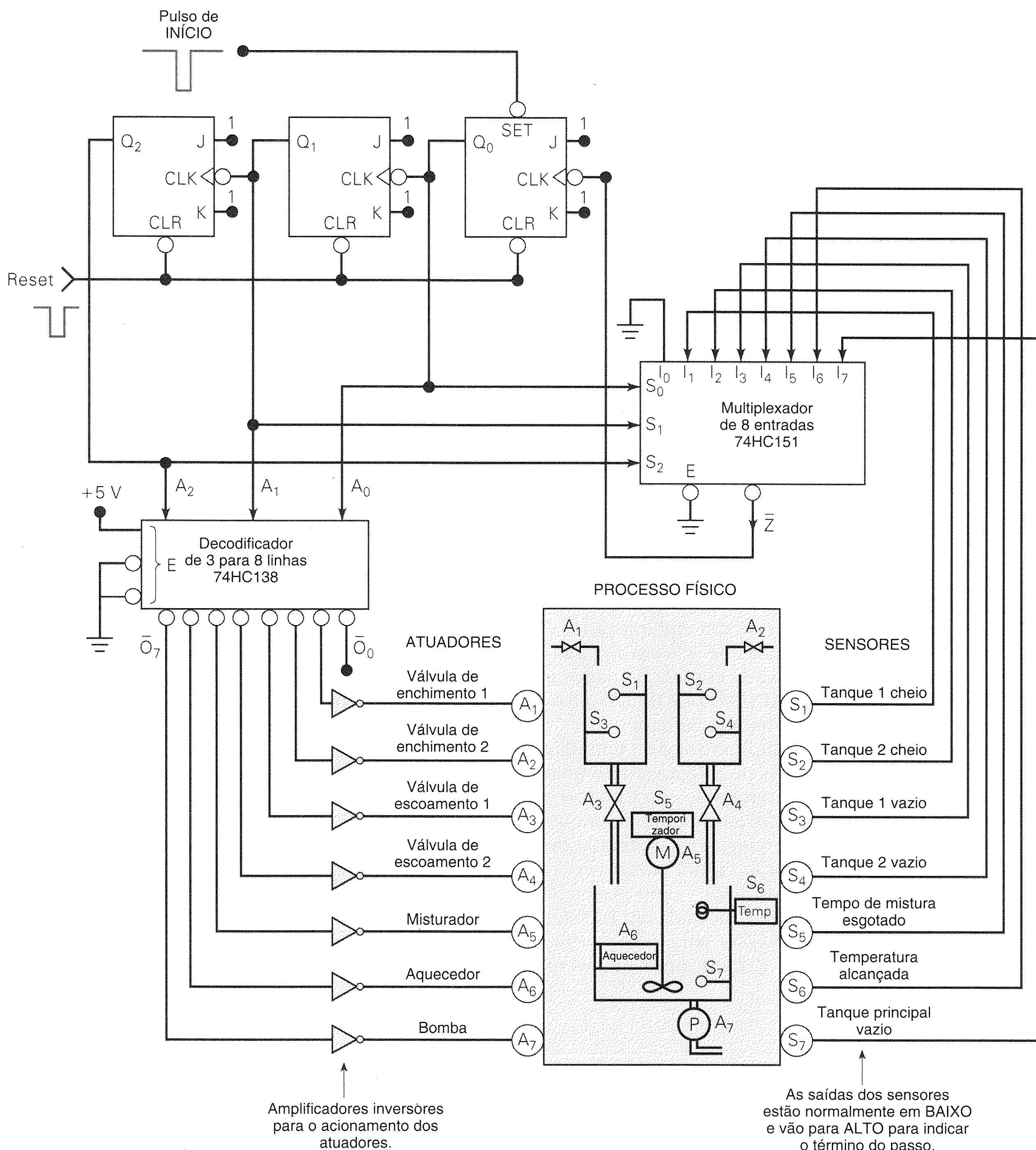


Fig. 9-28 Seqüenciador de controle de sete passos.

1. Inicialmente o contador é ressetado para o estado 000. As saídas do contador são levadas para as entradas de seleção do multiplexador e para as entradas do decodificador. Assim, a saída do decodificador $\overline{O}_0 = 0$ e todas as outras estão em 1, de modo que todas as entradas dos ATUADORES do processo estão em BAIXO. Todas as saídas dos SENSORES do processo iniciam em BAIXO. A saída do multiplexador $\overline{Z} = \overline{I}_0 = 1$, já que as entradas S são 000.
2. O pulso de INÍCIO começa a operação de seqüenciamento, setando o flip-flop Q_0 para ALTO, levando o contador para o estado 001. Isto faz com que a saída $\overline{O}_1$ do decodificador vá para BAIXO, ativando assim o atuador 1, que é o primeiro passo no processo (abrir a válvula de enchimento #1).
3. Algum tempo depois, a saída do SENSOR 1 vai para ALTO, indicando que o primeiro passo foi completado (a chave da bóia indica que o tanque está cheio). Este nível ALTO

aparece na entrada I_1 do multiplexador. Ele é invertido e alcança a saída $\overline{Z}$ pois o código de seleção do contador é 001.

4. A transição de $\overline{Z}$ para BAIXO é levada ao *CLK* do flip-flop Q_0. Esta transição negativa avança o contador para o estado 010.
5. A saída do decodificador $\overline{O}_2$ agora vai para BAIXO, ativando o atuador 2, que é o segundo passo do processo (abrir a válvula de enchimento #2). $\overline{Z}$ agora é igual a $\overline{I}_2$ (o código de seleção é 010). Como a saída do SENSOR 2 ainda está em BAIXO, $\overline{Z}$ vai para ALTO.
6. Quando o segundo passo do processo é concluído, a saída do SENSOR 2 vai para ALTO, produzindo um nível BAIXO em $\overline{Z}$ e avançando o contador para 011.
7. Esta mesma ação é repetida para cada um dos outros passos. Quando o sétimo passo é concluído, a saída do SENSOR 7 vai para ALTO, fazendo o contador ir de 111 para 000, onde permanece até que um outro pulso de INÍCIO recomece a seqüência.

Geração de Funções Lógicas

Multiplexadores podem ser usados para implementar funções lógicas diretamente a partir da tabela-verdade, sem necessidade de simplificação. Quando um multiplexador é usado para este propósito, as entradas de seleção são utilizadas como as variáveis lógicas, e cada entrada de dados é conectada permanentemente em ALTO ou BAIXO, conforme for necessário para satisfazer a tabela-verdade.

A Fig. 9-29 ilustra como um multiplexador de 8 entradas pode ser usado para implementar o circuito lógico de uma determinada tabela-verdade. As variáveis de entrada *A*, *B* e *C* são conectadas em S_0, S_1 e S_2, respectivamente, de modo que os níveis destas entradas determinam qual entrada de dados aparece na saída *Z*. De acordo com a tabela-verdade, *Z* deve estar em BAIXO quando *CBA* = 000. Assim, a entrada I_0 do multiplexador deve ser conectada em BAIXO. Analogamente, *Z* deve estar em BAIXO para *CBA* = 011, 100, 101 e 110, logo as entradas I_3, I_4, I_5 e I_6 também devem estar em BAIXO. As outras combinações de *CBA* devem produzir *Z* = 1, e portanto as entradas I_1, I_2 e I_7 do multiplexador são conectadas permanentemente em ALTO.

É fácil perceber que qualquer tabela-verdade de três variáveis pode ser implementada com este multiplexador de oito entradas. Este método de implementação freqüentemente é mais eficiente do que usar portas lógicas separadas. Por exemplo, a expressão da soma de produtos para a tabela-verdade na Fig. 9-29 é

$$Z = A\overline{B}\overline{C} + \overline{A}B\overline{C} + ABC$$

Ela *não pode* ser simplificada nem algebricamente nem pelo mapa de Karnaugh, e portanto a implementação com portas necessitaria de três INVERSORes e quatro portas NAND, perfazendo um total de dois CIs.

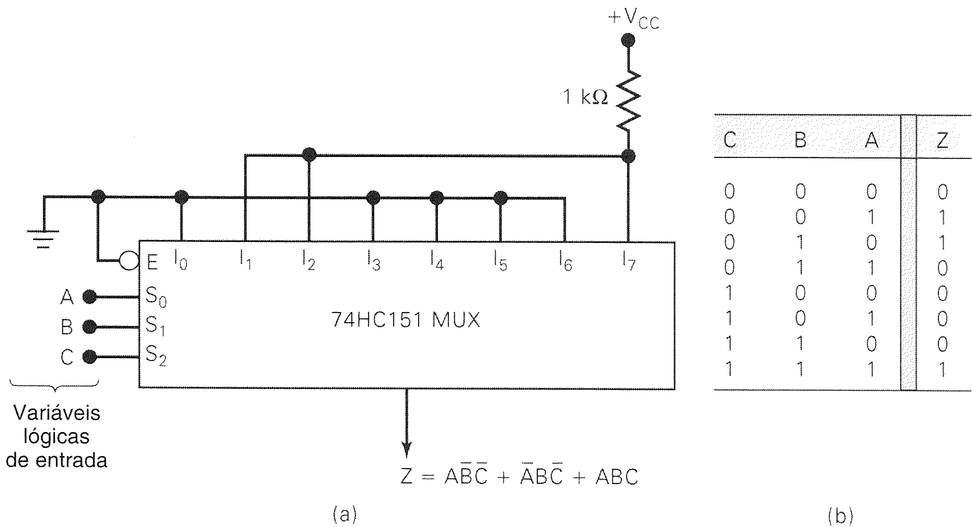
Existe um método ainda mais eficiente para usar multiplexadores na implementação de funções lógicas. Este método permite ao projetista usar um multiplexador com três entradas de seleção (por exemplo, um 74HC151) para implementar uma função lógica de *quatro variáveis*. Apresentaremos este método no Problema 9-35.

Questões de Revisão

1. Quais são algumas das principais aplicações dos multiplexadores?
2. *Verdadeiro ou falso:* Quando um multiplexador é usado para implementar uma função lógica, as variáveis lógicas são aplicadas nas suas entradas de dados.
3. Que tipo de circuito fornece as entradas de seleção quando um MUX é usado como um conversor paralelo-série?

9-9 DEMULTIPLEXADORES (DISTRIBUIDORES DE DADOS)

Um multiplexador recebe várias entradas e transmite *uma* delas para a saída. Um **demultiplexador (DEMUX)** reali-



C	B	A	Z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Fig. 9-29 Multiplexador utilizado para implementar uma função lógica descrita pela tabela-verdade.

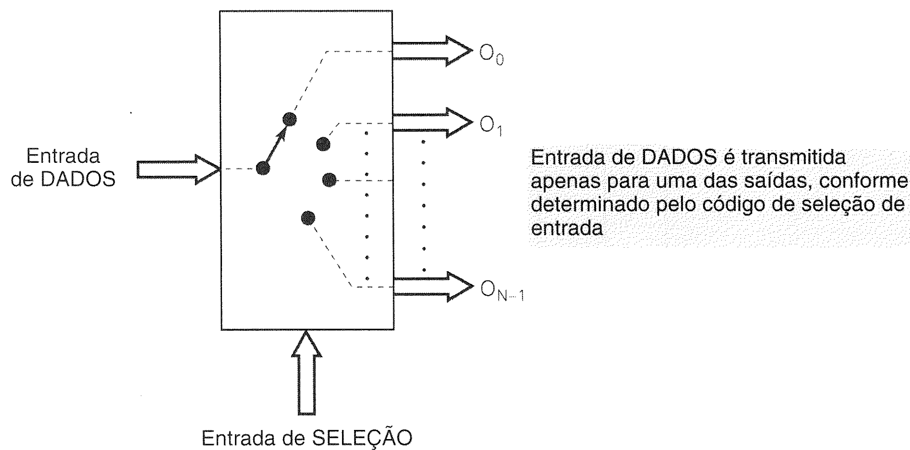


Fig. 9-30 Demultiplexador genérico.

za a operação inversa: ele recebe uma única entrada e a distribui por várias saídas. A Fig. 9-30 mostra o diagrama funcional para um demultiplexador digital. As setas largas para as entradas e saídas podem representar uma ou mais linhas. O código de seleção de entrada determina para qual saída a entrada de DADOS será transmitida. Em outras palavras, o demultiplexador recebe uma fonte de dados e seletivamente a distribui para 1 entre *N* canais de saída, como se fosse uma chave de várias posições.

Demultiplexador de 1 para 8 Linhas

A Fig. 9-31 mostra o diagrama lógico para um demultiplexador que distribui uma linha de entrada para oito linhas de saída. A única linha de entrada de dados *I* é conectada em todas as portas AND, mas somente uma destas portas será habilitada pelas linhas de SELEÇÃO de entrada. Por exemplo, com $S_2S_1S_0$ = 000, apenas a porta AND 0 estará habilitada, e a entrada de dados *I* aparecerá na saída O_0. Outros códigos de SELEÇÃO fazem a entrada *I* alcançar as outras saídas. A tabela-verdade resume a operação.

O circuito demultiplexador da Fig. 9-31 é bastante similar ao circuito decodificador 3 para 8 na Fig. 9-2, exceto que uma quarta entrada (*I*) foi adicionada em cada porta. Foi ressaltado anteriormente que muitos decodificadores possuem uma entrada de HABILITAÇÃO, que é uma entrada extra adicionada às portas dos decodificadores. Portanto, este tipo de chip decodificador pode ser usado como um demultiplexador, com o código binário da entrada (por exemplo, *A*, *B* e *C* na Fig. 9-2) servindo como entradas de SELEÇÃO e a entrada de HABILITAÇÃO servindo como a entrada de dados *I*. Por esta razão, os fabricantes de CIs freqüentemente chamam este dispositivo de *decodificador/demultiplexador*, e ele pode ser usado para ambas as funções.

Estudamos anteriormente como o 74LS138 é usado como um decodificador 3 para 8. A Fig. 9-32 mostra como ele pode ser utilizado como um demultiplexador. A entrada de habilitação $\overline{E}_1$ é usada como a entrada de dados *I*, enquanto as outras duas entradas de habilitação são mantidas em seus estados ativos. As entradas $A_2A_1A_0$ são utilizadas como o código de seleção. Para ilustrar a operação, vamos admitir que as entradas de seleção são 000. Com este código de entrada, a única saída que pode ser ativada é $\overline{O}_0$, enquanto todas as outras saídas estão em ALTO. $\overline{O}_0$ vai para BAIXO somente se $\overline{E}_1$ for para BAIXO, e ficará em ALTO se $\overline{E}_1$ for ALTO. Em outras palavras, *$\overline{O}_0$ vai seguir o sinal em $\overline{E}_1$* (isto é, a entrada de dados, *I*) enquanto todas as outras saídas permanecem em ALTO. Da mesma maneira, um código de seleção diferente aplicado em $A_2A_1A_0$ vai fazer a saída correspondente seguir a entrada de dados, *I*.

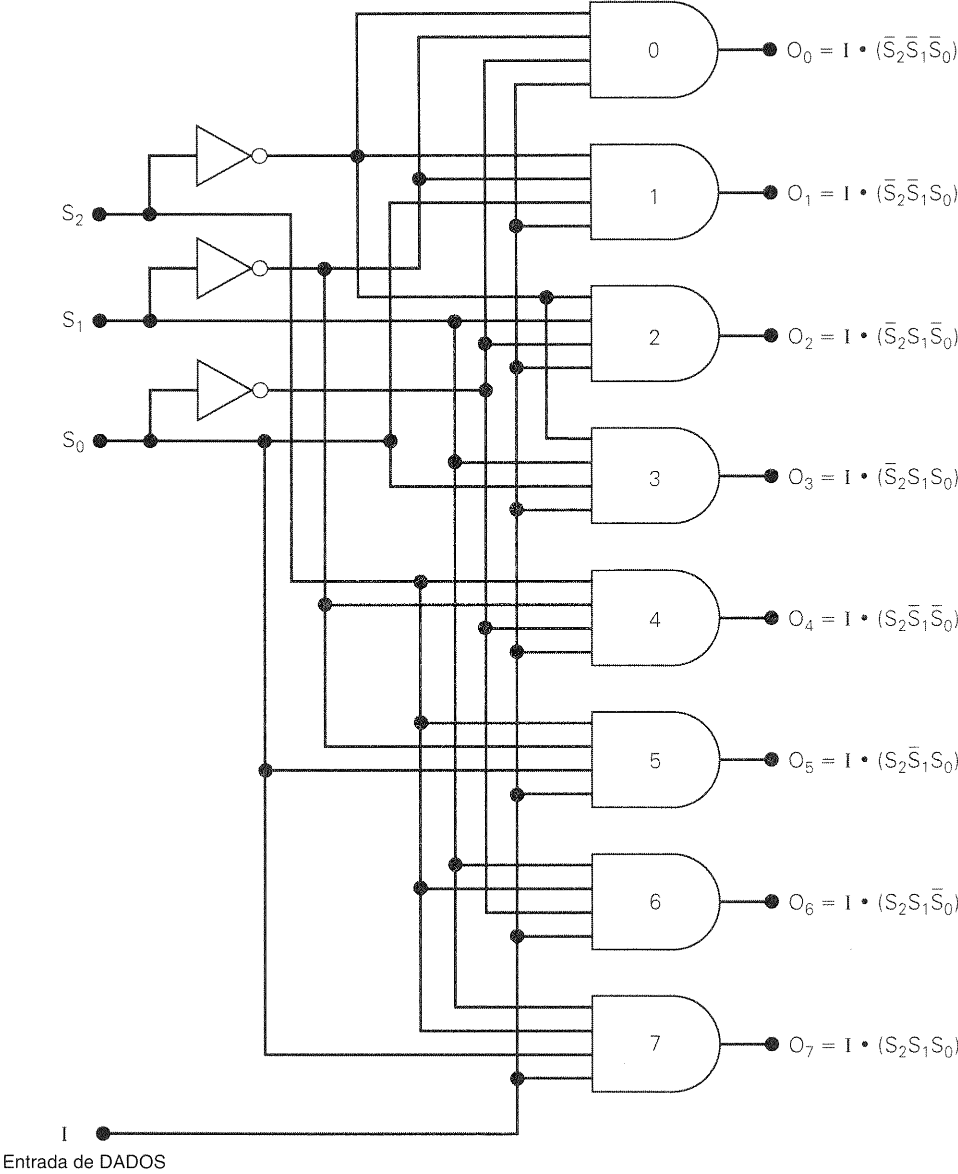
A Fig. 9-32(b) mostra as formas de onda típicas para o caso em que $A_2A_1A_0$ = 000 selecionam a saída $\overline{O}_0$. Para este caso, o sinal de dados aplicado em $\overline{E}_1$ será transmitido para $\overline{O}_0$, e todas as outras saídas permanecerão em seus estados inativos em ALTO.

Demultiplexador de Clock

Muitas aplicações do princípio de demultiplexação são possíveis. A Fig. 9-33 mostra o demultiplexador 74LS138 sendo usado como um *demultiplexador de clock*. Sob o controle das linhas de SELEÇÃO, o sinal de clock é roteado para o destino desejado. Por exemplo, com $S_2S_1S_0$ = 000, o sinal de clock aplicado a *I* vai aparecer na saída $\overline{O}_0$. Com $S_2S_1S_0$ = 101, clock aparecerá em $\overline{O}_5$.

Sistema de Monitoração de Segurança

Considere o caso de um sistema de monitoração de segurança em uma instalação industrial onde o estado aberto ou fechado de várias portas de acesso deve ser monitorado. Cada porta controla o estado de uma chave e é necessário mostrar o estado de cada chave através de LEDs que estão montados num painel remoto de monitoração na sala do pessoal de segurança. Uma maneira de fazer isto seria levar o sinal de cada porta para um LED no painel. Isto exigiria a instalação de uma grande quantidade de fios por uma distância considerável. Uma solução mais adequada, que reduziria a quantidade de fios para o painel de monitoração, utilizaria uma combinação multiplexador/demultiplexador. A Fig. 9-34 mostra um sistema que pode tratar oito portas,



Código de SELEÇÃO			SAÍDAS							
S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0

Nota: **I** é a entrada de dados

Fig. 9-31 Demultiplexador de 1 para 8 linhas.

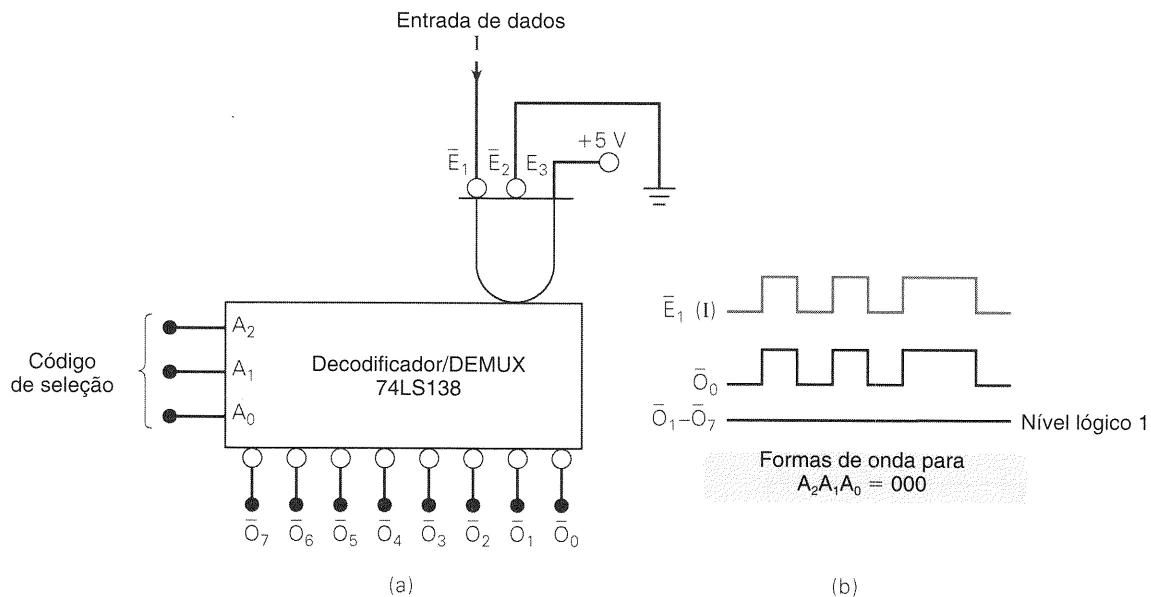


Fig. 9-32 (a) O decodificador 74LS138 pode funcionar como um demultiplexador com $\overline{E}_I$ usado como a entrada de dados. (b) Formas de onda típicas para o código de seleção de $A_2A_1A_0 = 000$ mostram que $\overline{O}_0$ é idêntico à entrada de dados I em $\overline{E}_I$.

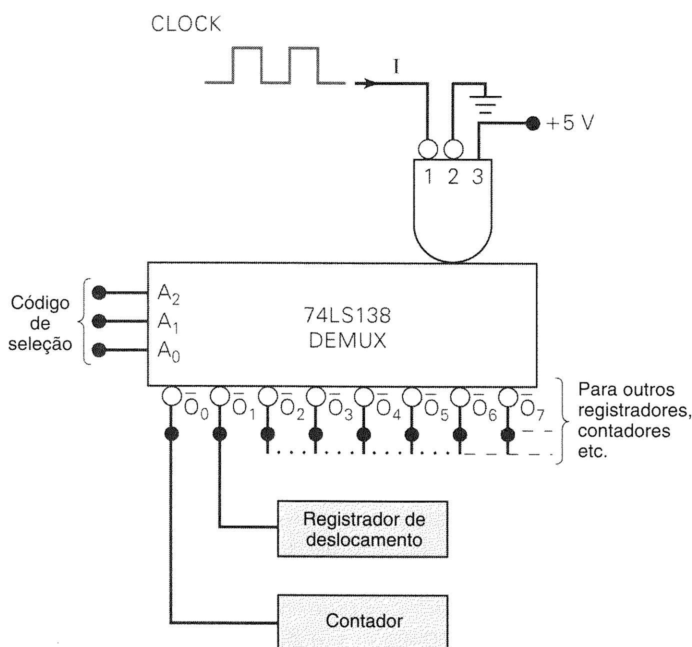


Fig. 9-33 Um demultiplexador de clock transmite o sinal de clock para o destino determinado pelo código de seleção de entrada.

mas a idéia básica pode ser expandida para qualquer número.

EXEMPLO 9-11

Examine a Fig. 9-34 cuidadosamente e descreva sua operação completa.

Solução

As oito chaves das portas são as entradas de dados do MUX; elas produzem um nível ALTO quando uma porta estiver aberta e um nível BAIXO quando estiver fechada. O contador de módulo 8 fornece as entradas de seleção do MUX e também do DEMUX do painel remoto de monitoração. Cada saída do DEMUX está conectada a um LED indicador que estará ligado quando a saída estiver em nível BAIXO. Os pulsos de clock aplicados ao contador fazem as entradas de seleção variarem por todos os estados possíveis desde 000 até 111. Para cada número da contagem, o estado da chave da porta de mesmo número é invertido pelo MUX e enviado para a saída $\overline{Z}$. De lá ele é transmitido para a entrada do DEMUX, que o passa para a saída correspondente ao mesmo número.

Por exemplo, digamos que o contador esteja na contagem 110 (6). Enquanto o contador estiver neste estado, admitamos que a porta 6 esteja fechada. O nível BAIXO em I_6 vai passar pelo MUX e ser invertido para produzir um nível ALTO em $\overline{Z}$. Este ALTO vai passar pelo DEMUX até a saída $\overline{O}_6$, de modo que o LED 6 ficará apagado, indicando que a porta 6 está fechada. Agora vamos imaginar que a porta 6 seja aberta. Um nível BAIXO aparecerá em $\overline{Z}$ e em $\overline{O}_6$, logo o LED 6 acenderá para sinalizar que a porta 6 está aberta. Naturalmente, todos os outros LEDs ficam apagados durante este tempo, já que $\overline{O}_6$ é a única saída ativa.

Conforme o contador avança por seus estados desde 000 até 111, os LEDs seqüencialmente indicam o estado das oito portas. Se todas as portas estiverem fechadas, nenhum dos LEDs acende, mesmo quando a saída correspondente do DEMUX for selecionada. Se uma porta for aberta, seu LED acenderá apenas durante o intervalo de tempo que o contador ficar com a contagem apropriada; ele estará apagado para todas as outras contagens. Assim, o LED ficará piscan-

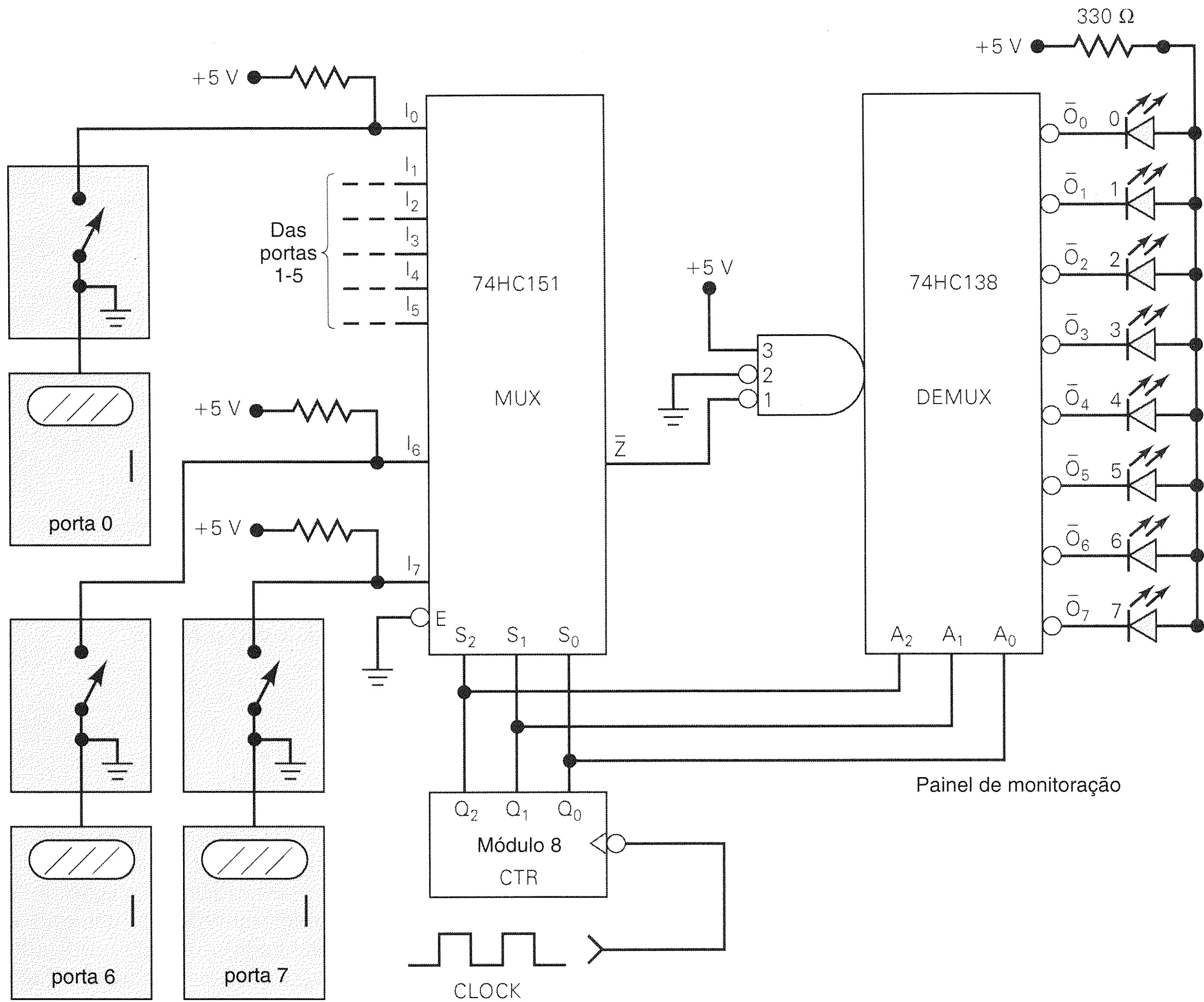


Fig. 9-34 Sistema de monitoração de segurança.

do se a porta estiver aberta. A taxa na qual o LED pisca pode ser ajustada alterando-se a freqüência do clock.

Repare que existem apenas quatro linhas de sinal indo dos circuitos que tratam os sensores das portas para o painel remoto de monitoração: a saída $\overline{Z}$ e as três linhas de seleção. Isto economiza quatro linhas quando comparado com a alternativa de ter uma linha por porta. A combinação MUX/DEMUX é usada para transmitir o estado de cada porta para seu LED, um em cada instante (serialmente), em lugar de todos de uma só vez (paralelamente).

Sistema Síncrono de Transmissão de Dados

A Fig. 9-35 mostra o diagrama lógico para um sistema síncrono de transmissão de dados que é utilizado para transmitir serialmente quatro palavras de dados de quatro bits de um transmissor para um receptor remoto. Vamos analisar o circuito transmissor primeiro. As palavras estão armazenadas em registradores *A*, *B*, *C* e *D* que estão conectados como registradores de deslocamento circulares com uma entrada de clock comum. Cada registrador vai deslocar para a direita, a cada transição de subida dos pulsos de deslocamento da porta AND 2. O LSB de cada registrador está conectado em uma entrada de dados do multiplexador de quatro entradas.

Os dois contadores de módulo 4 controlam a transmissão do conteúdo do registrador de dados para a saída *Z* do multiplexador. O *contador de palavras* seleciona o registrador de dados que vai aparecer em *Z*. Conforme este contador avança desde 00 até 11, os dados de cada registrador aparecem seqüencialmente em *Z*. O *contador de bits* garante que quatro bits de dados de cada registrador são transmitidos, através do multiplexador, antes do avanço para o próximo registrador. O contador de bits avança a cada pulso de deslocamento, de modo que depois de quatro pulsos ele recicla para 00. A descida da saída Q_1 do contador de bits provoca o incremento do contador de palavras para a próxima contagem, para selecionar o próximo registrador de dados para a transmissão. Desta maneira, o conteúdo de cada registrador de dados é transmitido para *Z*, um bit de cada vez, iniciando com o registrador *A* (para $S_1S_0 = 00$) e passando por cada registrador, conforme o contador de palavras avança uma contagem a cada quatro pulsos de deslo-

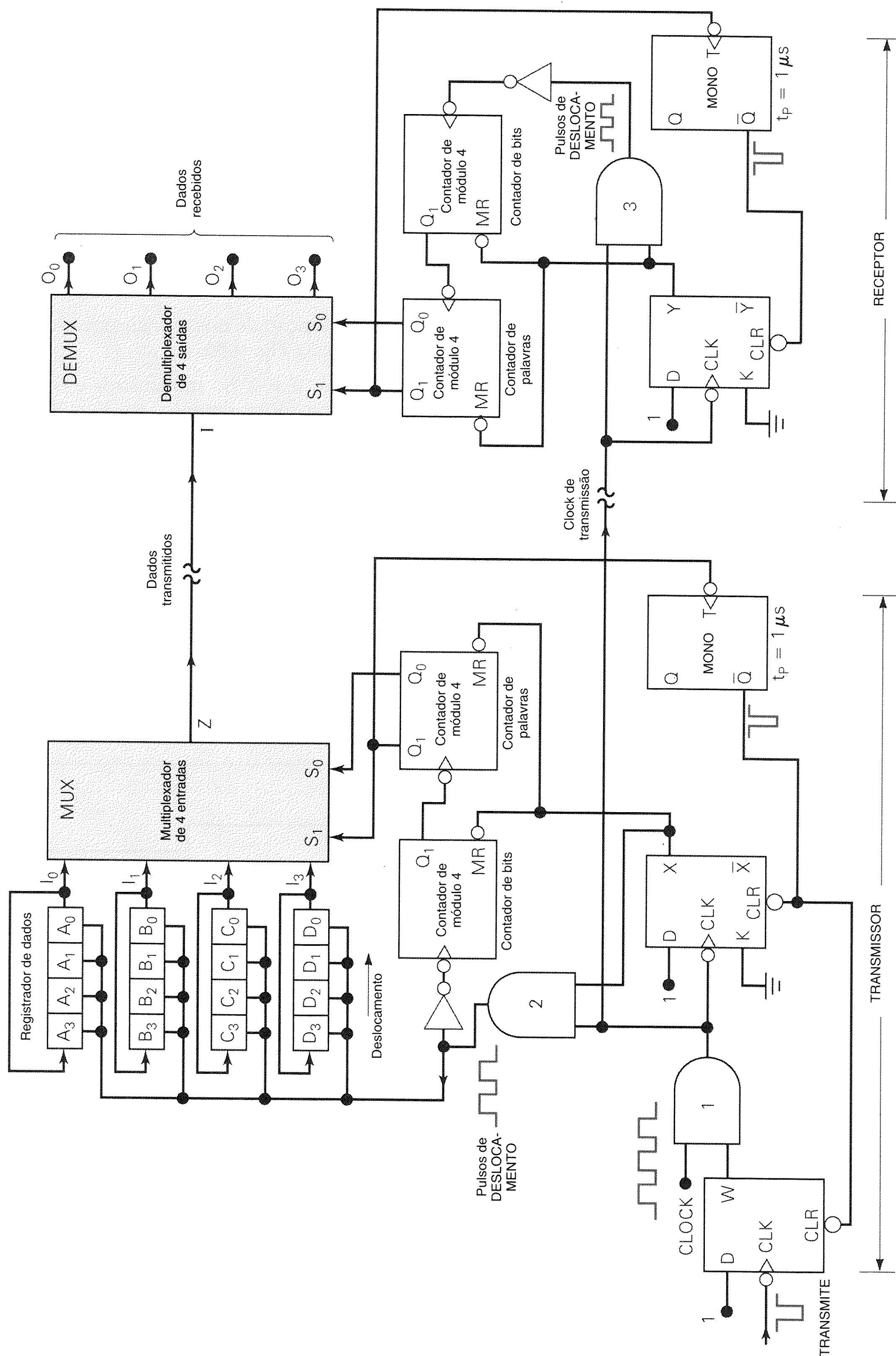


Fig. 9-35 Sistema síncrono de transmissão de dados.

camento. Portanto, o sinal Z contém 16 bits de dados seriais, 4 bits de cada registrador. Estes dados são *multiplexados por divisão do tempo*, pois quatro conjuntos de dados diferentes aparecem na mesma linha de saída em momentos diferentes.

O processo de transmissão é controlado pelos dois flip-flops D, pelas portas AND 1 e 2 e pelo monoestável. A operação desta lógica de controle será descrita mais adiante.

O Receptor

O circuito receptor contém um demultiplexador de 1 para 4 que recebe o sinal Z do multiplexador do transmissor e o *demultiplexa*, isto é, ele separa os quatro diferentes conjuntos de dados e os distribui para quatro saídas diferentes, de modo que os dados oriundos do registrador A aparecem serialmente em O_0, os dados do registrador B em O_1, e assim por diante. O resultado final é quase o mesmo de se ter cada registrador transmissor de dados conectado diretamente à sua saída correspondente no receptor, exceto que os dados são enviados por um registrador de cada vez através da linha de transmissão serial.

Os contadores de módulo 4 no receptor têm a mesma função que seus correspondentes no transmissor. O contador de palavras seleciona qual das saídas do demultiplexador estará recebendo os dados, e o contador de bits permite que quatro bits de dados alcancem cada saída antes do avanço do contador de palavra para o seu próximo estado. As funções do FF, do MONO e da porta AND estão descritas a seguir.

Operação Completa

Deve ficar claro que, para esta transmissão de dados funcionar adequadamente, tem que existir algum meio de sincronizar a seleção das entradas do multiplexador no transmissor com a seleção das saídas do demultiplexador no receptor. Vamos analisar um ciclo de operação completo para verificar como esta sincronização é conseguida. Para isto, vamos considerar os seguintes valores para os registradores:

$[A] = 0110 \quad [B] = 1001 \quad [C] = 1011 \quad [D] = 0100$

Conforme a realização de nossa análise, vamos consultar as formas de onda na Fig. 9-36:

1. Os flip-flops W e X no transmissor e o flip-flop Y no receptor estão normalmente em nível BAIXO. As saídas em BAIXO de X e Y mantêm ambos os conjuntos de contadores no estado 0. O nível BAIXO em W não permite que os pulsos de CLOCK ultrapassem a porta AND 1.
2. Com ambos os contadores de palavras em 00, o nível BAIXO em A_0 passa através do multiplexador para Z, e daí para a saída O_0 do demultiplexador. Todas as outras saídas do demultiplexador ficam em BAIXO, já que elas não estão sendo selecionadas.

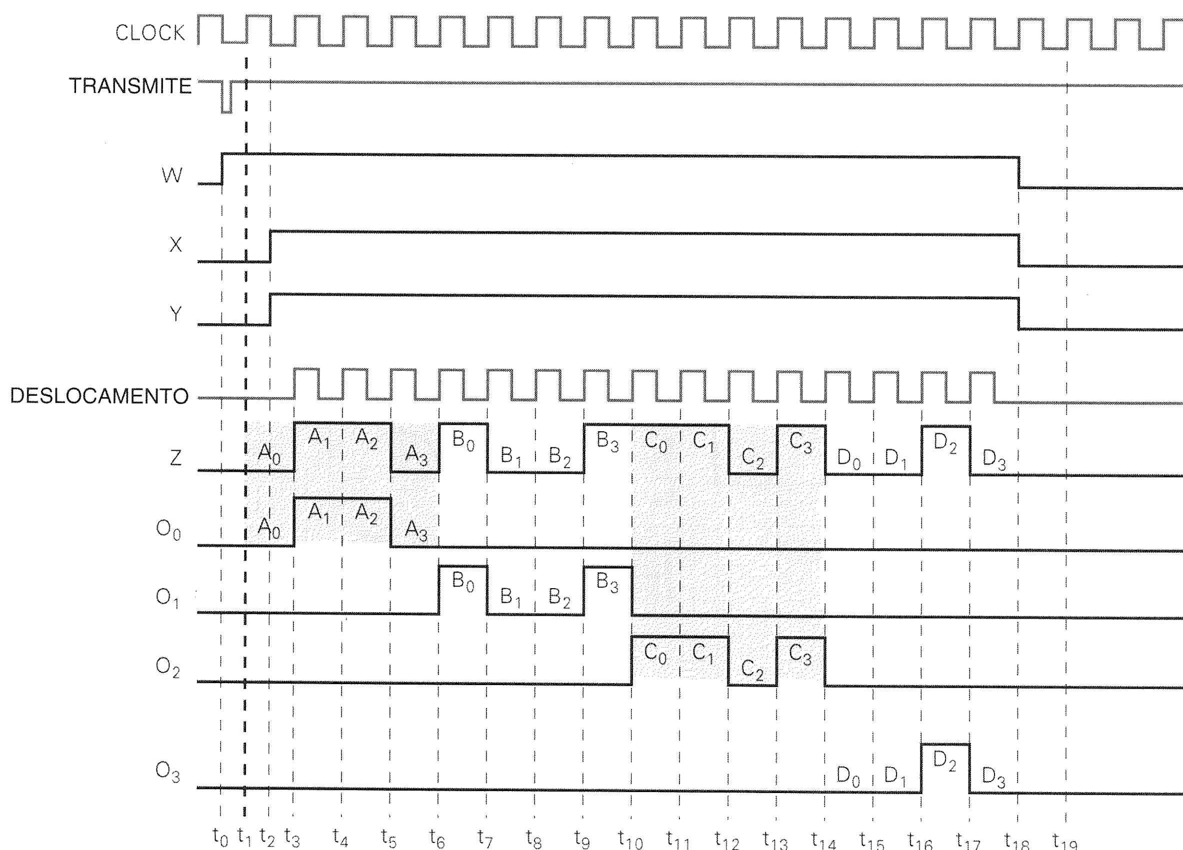


Fig. 9-36 Formas de onda durante um ciclo de transmissão completo.

3. Esta é a situação antes do instante t_0. Em t_0, o pulso TRANSMITE seta $W = 1$ para habilitar a porta AND 1 a passar os pulsos de CLOCK. Estes pulsos de CLOCK também se tornam o clock de transmissão que é enviado para o receptor junto com os dados transmitidos.
4. A descida do primeiro pulso de CLOCK, após a porta AND 1, seta X e Y para ALTO no instante t_2. Isto remove o reset dos contadores e também habilita as portas AND 2 e 3 a passarem os pulsos de CLOCK a partir de t_3. Os pulsos na saída da porta AND 2 são os pulsos de deslocamento. Os pulsos de saída da porta AND 3 são exatamente os mesmos que os pulsos de deslocamento na saída da porta AND 2.
5. As subidas dos pulsos de deslocamento em t_3, t_4 e t_5 deslocam A_1, A_2 e A_3 para o multiplexador, para Z e para a saída O_0 do demultiplexador. Estas três subidas também são contadas por ambos os contadores de bits.
6. A subida em t_6 traz os registradores de volta aos seus dados originais e recicla os contadores de bits para 00. A descida de Q_1 dos contadores de bits incrementam os contadores de palavras do transmissor e do receptor para 01, para selecionarem I_1 e O_1, respectivamente. Assim, o nível ALTO de B_0 passa através do multiplexador para o demultiplexador e para a saída O_1.
7. Os pulsos de DESLOCAMENTO em t_7, t_8 e t_9 deslocam B_1, B_2 e B_3 para o multiplexador e para a saída O_1. Em t_{10}, os contadores de bits reciclam e incrementam os contadores de palavras para 10, para selecionarem I_2 e O_2. Isto coloca o nível ALTO de C_0 em Z e em O_2.
8. Os pulsos de DESLOCAMENTO em t_{11}, t_{12} e t_{13} deslocam C_1, C_2 e C_3 para o multiplexador e para a saída O_2. Em t_{14}, os contadores de bits reciclam e incrementam os contadores de palavras para 11, para selecionarem I_3 e O_3. Isto coloca o nível BAIXO de D_0 em Z e em O_3.
9. Os pulsos de DESLOCAMENTO em t_{15}, t_{16} e t_{17} deslocam D_1, D_2 e D_3 para o multiplexador e para a saída O_3. Em t_{18}, os contadores de bits reciclam e incrementam os contadores de palavras para 00. A descida de Q_1 dos *contadores de palavras* dispara seus respectivos monoestáveis para produzirem um pulso estreito para limpar os FFs W, X e Y. Com estes FFs em BAIXO, todos os pulsos de CLOCK e de DESLOCAMENTO são inibidos, e todos os contadores permanecem no estado 0.
10. O circuito retornou as condições iniciais. Nenhum dado será transmitido até que o próximo pulso TRANSMITE ocorra.

As formas de onda de Z e de O_0 a O_3 mostram como os dados dos registradores são multiplexados no sinal Z, e então demultiplexados, de modo que cada saída receba os dados corretos.

Questões de Revisão

1. Explique a diferença entre um DEMUX e um MUX.
2. *Verdadeiro ou falso:* O circuito para um DEMUX é basicamente o mesmo de um decodificador.
3. Para o sistema da Fig. 9-34, o que o guarda de segurança vai ver no painel de monitoração quando todas as portas estiverem abertas?

9-10 MAIS SOBRE A SIMBOLOGIA IEEE/ANSI*

Vamos estudar agora os símbolos IEEE/ANSI para os circuitos integrados MUX e DEMUX que temos utilizado. Primeiramente consideremos o símbolo para o 74151 mostrado na Fig. 9-37(a). O rótulo $G\frac{0}{7}$ dentro do bloco indica a dependência AND entre as entradas de seleção e cada uma das entradas de dados de 0 a 7. Em outras palavras, com cada entrada de dados é feita uma operação AND com uma combinação específica de entradas de seleção, e, quando esta combinação ocorre, a entrada de dados é roteada para a saída.

A Fig. 9-37(b) é o símbolo IEEE/ANSI para o multiplexador 74157. Lembre-se, da Fig. 9-25, que este CI contém quatro MUXes de duas entradas que operam de modo idêntico. O bloco de controle comum mostra que as entradas de habilitação e seleção são comuns para todos os MUXes. A notação G1 na entrada de seleção e os rótulos $\bar{1}$ e 1 nas entradas de dados indicam a dependência AND entre a entrada de seleção e as entradas de dados. O rótulo $\bar{1}$ para I_{0a} significa que esta entrada será roteada para a saída Z_a somente quando a entrada de seleção for 0. Analogamente, o 1 na entrada I_{1a} indica que esta entrada será roteada para Z_a somente quando a entrada de seleção for 1. Os outros três MUXes operam do mesmo modo.

EXEMPLO 9-12

O símbolo IEEE/ANSI para o CI 74150 está mostrado na Fig. 9-37(c). Qual é a função deste dispositivo?

Solução

O 74150 é um multiplexador de 16 entradas. Note as quatro entradas de seleção necessárias para selecionar uma das 16 entradas para transmissão para a saída. Repare também que a saída é invertida, de modo que ela será o inverso da entrada de dados selecionada.

A Fig. 9-38 é o símbolo IEEE/ANSI para o DEMUX 74LS138/HC138. Compare-o com o símbolo na Fig. 9-17(c), onde o mesmo CI é utilizado como um decodificador.

9-11 MAIS PESQUISA DE FALHAS

Vamos apresentar mais três exemplos para ilustrar o processo de observação/raciocínio que é um passo inicial de grande importância para a depuração. Para cada caso, tente determinar a falha do circuito antes de ver as soluções.

EXEMPLO 9-13

Considere o circuito da Fig. 9-26. Um teste realizado neste circuito produziu os resultados mostrados na Tabela 9-2. Qual é a falha provável do circuito?

*Esta seção pode ser omitida sem prejudicar a continuidade do restante do livro.

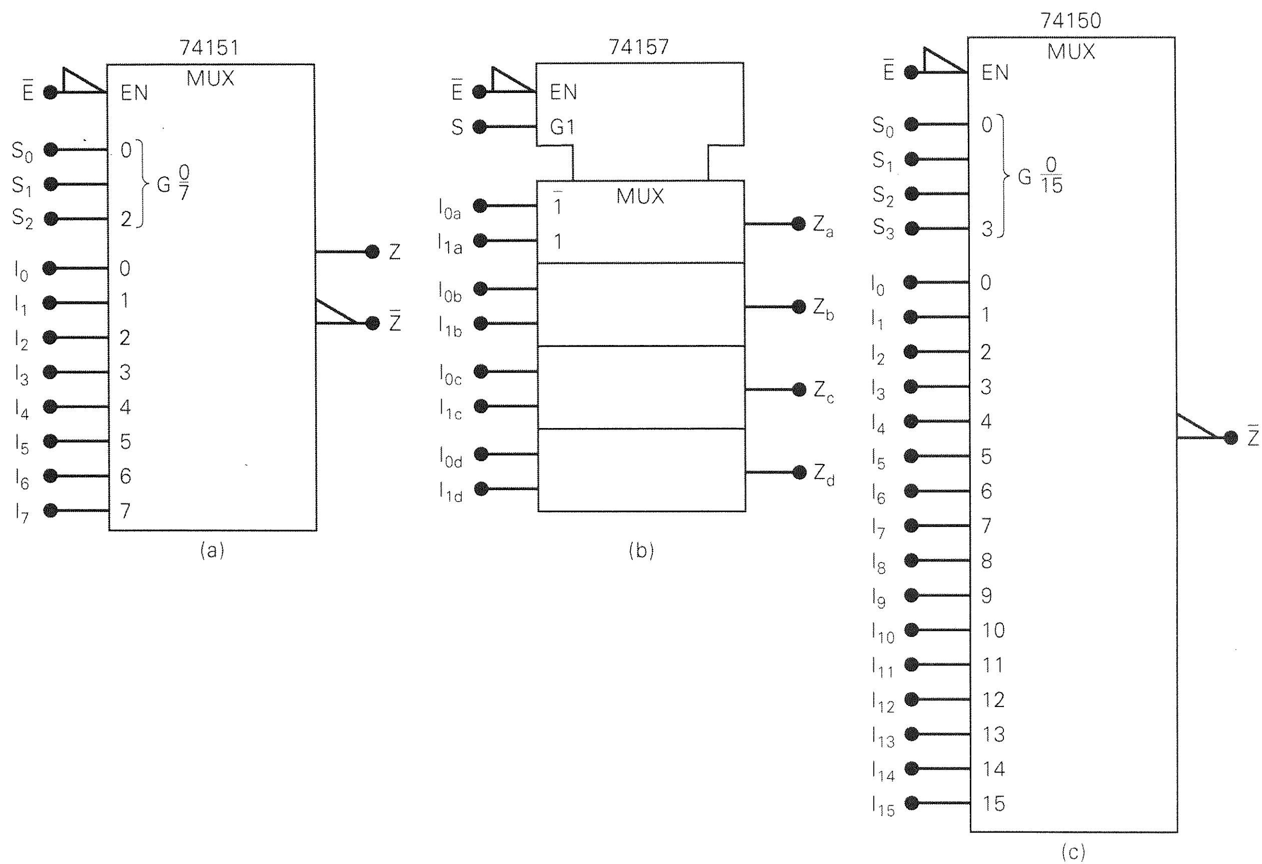


Fig. 9-37 Símbolos IEEE/ANSI para vários multiplexadores.

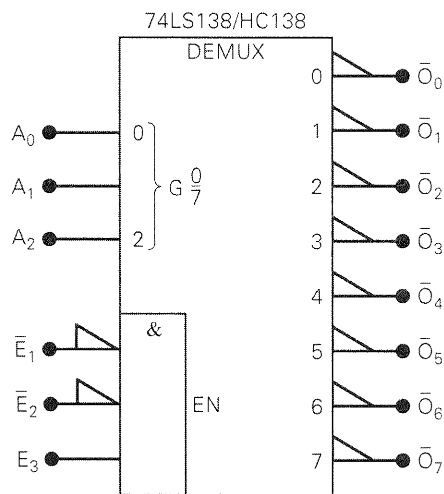


Fig. 9-38 Símbolo IEEE/ANSI para o DEMUX 74LS138.

Solução

Em cada um dos casos de teste, o display do contador 1 coincide com o valor real da sua contagem. Isto indica que as entradas I_1, todas as saídas dos MUXes e ambos os displays provavelmente estão funcionando corretamente. Por outro lado, cada caso de teste mostra que o dígito das *dezenas* do contador 2 é apresentado corretamente mas o dígito das *unidades* não. Isto poderia significar a existência de uma falha

TABELA 9-2

		Contagem Real	Valor no Display
Caso 1	Contador 1	25	25
	Contador 2	37	35
Caso 2	Contador 1	49	49
	Contador 2	72	79
Caso 3	Contador 1	96	96
	Contador 2	14	16

TABELA 9-3

	Unidades Reais	Unidades Mostradas
Caso 1	0111 (7)	0101 (5)
Caso 2	0010 (2)	1001 (9)
Caso 3	0100 (4)	0110 (6)

em algum lugar entre a saída da seção de unidades do contador 2 e as entradas I_0 do MUX das unidades. Devemos comparar os padrões de bits do valor real com o valor mostrado para as unidades do contador 2 (Tabela 9-3). A idéia é procurar por algo como um bit que não muda (sempre em BAIXO ou ALTO) ou dois bits que estejam trocados (conexões erradas). Os dados da Tabela 9-3 não revelam qualquer padrão óbvio.

Observando novamente os resultados registrados dos testes, vemos que o dígito das unidades do contador 2 mostrado é sempre o mesmo que o dígito das unidades do contador 1. Este sintoma provavelmente é o resultado de um nível lógico ALTO constante na entrada de seleção do MUX das unidades, pois ele passaria continuamente o dígito das unidades do contador 1 para as suas saídas. Este nível lógico ALTO constante na entrada de seleção provavelmente é devido a uma ligação aberta entre a entrada de seleção do MUX das dezenas e a entrada de seleção do MUX das unidades. Tal efeito não poderia ser causado por um curto-circuito para V_{CC}, tendo em vista que isto manteria a entrada de seleção do MUX das dezenas constantemente em ALTO e sabemos que o MUX das dezenas está funcionando.

EXEMPLO 9-14

O sistema de monitoração de segurança da Fig. 9-34 é testado e os resultados estão registrados na Tabela 9-4. Quais as possíveis falhas que poderiam provocar estes resultados?

Solução

Novamente, os dados devem ser examinados para verificar se existe algum padrão que poderia ajudar a restringir a procura da falha a uma área menor do circuito. Os dados na Tabela 9-4 revelam que os LEDs corretos piscam para as portas de 4 a 7 abertas. Também mostram que para as portas de 0 a 3 abertas o número do LED que pisca é *quatro* a mais do que o número da porta, e os LEDs de 0 a 3 sempre estão apagados. Isto é provavelmente causado por um nível lógico ALTO constante em A_2, o MSB da entrada de seleção do DEMUX, pois isto sempre faria o código de seleção ser 4 ou maior, e somaria 4 aos códigos de seleção entre 0 e 3.

Assim, temos duas possibilidades: A_2 está de algum modo em curto com V_{CC}, ou existe uma conexão aberta em A_2. Um pouco mais de raciocínio elimina a primeira opção, já que ela também significaria que o S_2 do MUX estaria fixo em ALTO. Se isto fosse verdade, então o estado das portas 0 a 3 não passaria pelo MUX e pelo DEMUX. Sabemos que isto não é verdade porque os dados mostram que, quando qualquer uma destas portas está aberta, ela afeta uma das saídas do DEMUX.

TABELA 9-4

Condição	LEDs
Todas as portas fechadas	Todos os LEDs apagados
Porta 0 aberta	LED 4 piscando
Porta 1 aberta	LED 5 piscando
Porta 2 aberta	LED 6 piscando
Porta 3 aberta	LED 7 piscando
Porta 4 aberta	LED 4 piscando
Porta 5 aberta	LED 5 piscando
Porta 6 aberta	LED 6 piscando
Porta 7 aberta	LED 7 piscando

EXEMPLO 9-15

Suponha que o sistema síncrono de transmissão de dados da Fig. 9-35 esteja funcionando mal, do seguinte modo: a forma de onda Z está correta, mas a forma de onda de O_0 está idêntica à forma de onda de Z a qualquer momento enquanto as outras saídas estão constantemente em BAIXO. Admita que o circuito receptor está soldado numa placa de circuito impresso sem soquetes.

Solução

Observação/análise deve ser utilizada para determinar as possíveis causas. Dividir e conquistar deve ser usado para isolar o problema. A causa mais óbvia parece ser que S_1 e S_0 do DEMUX estão sempre em BAIXO conforme os dados são transmitidos. Se *não* for isto, então o DEMUX *deve* estar defeituoso. Utilizando uma ponta de prova lógica, eliminaríamos esta possibilidade primeiro, pois é um teste simples que poderia isolar o problema de todos os outros circuitos. Admitindo que encontrássemos S_1 e S_0 *sempre* em BAIXO, existem diversas causas possíveis para este sintoma. Vamos listá-las:

1. S_0 do MUX ou Q_0 do contador de palavras poderiam estar em curto com a terra, impedindo que o contador incrementasse.
2. O contador de palavras poderia estar com defeito (linha de *CLK* aberta ou em curto, *MR* em curto com a terra, ou um defeito interno).
3. O contador de bits poderia estar com defeito (*CLK* ou Q_1 aberto ou em curto, *MR* em curto com a terra, ou um defeito interno).
4. O INVERSOR ou a porta AND poderiam estar com defeito (entradas ou saídas abertas ou em curto, ou defeitos internos).
5. O FF *Y* poderia estar defeituoso (*D*, *CLR* ou *Y* em curto com a terra, ou defeitos internos).
6. O monoestável poderia estar com defeito (disparo em curto ou aberto, Q_0 em curto com a terra, falha no circuito *RC* conectado ao monoestável, ou defeitos internos).
7. A linha de clock de transmissão pode estar aberta entre o transmissor e o receptor.

Tendo em vista que a troca de todos os componentes significa dessoldar chips de 14 e 16 pinos, não desejamos

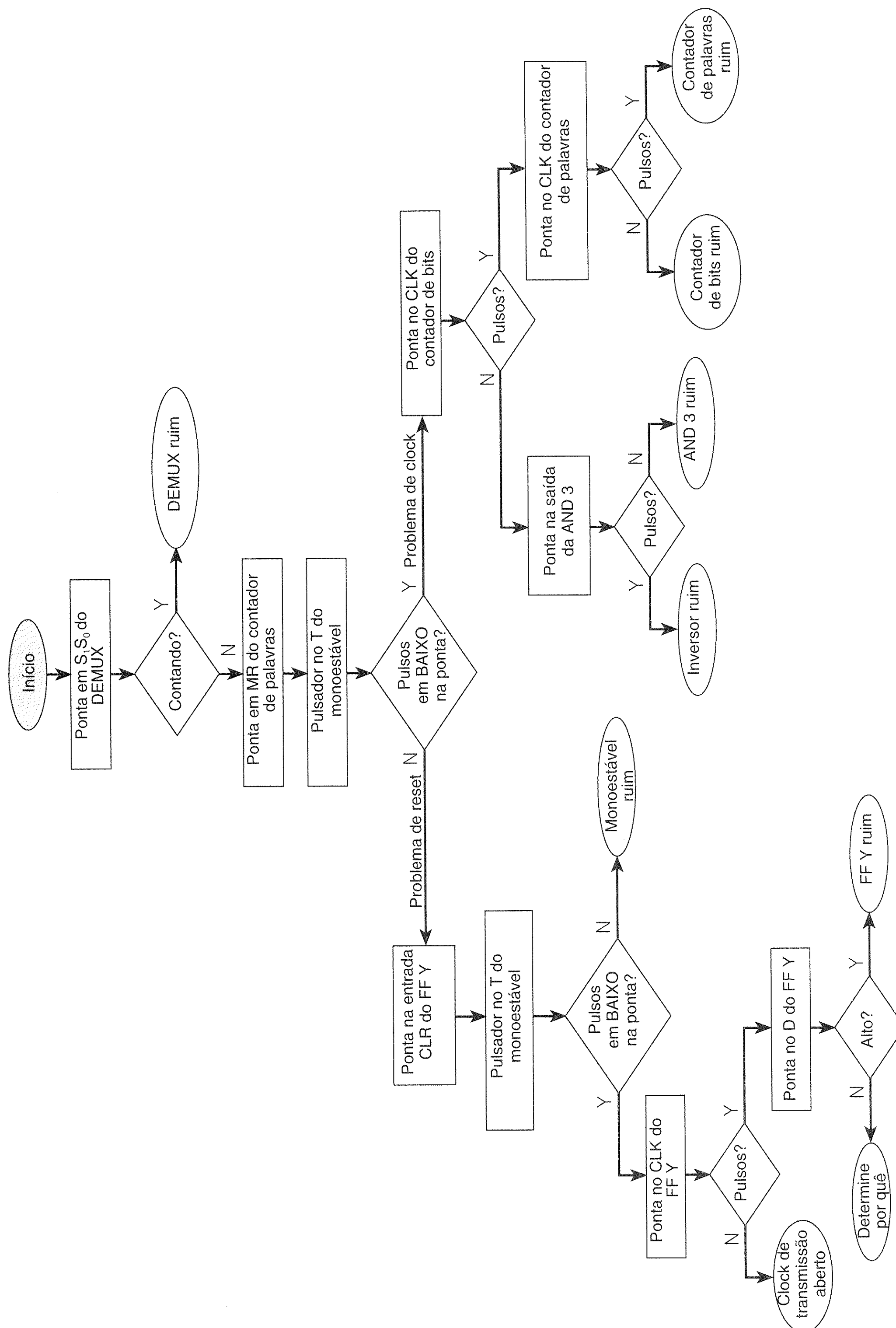


Fig. 9-39 Exemplo 9-15: um diagrama de depuração.

empregar a estratégia da "força bruta" na depuração. Temos que encontrar um modo de eliminar possíveis causas de falha pela estratégia de dividir e conquistar. Começando pelo contador de palavras, decidimos que ou o clock não está sendo pulsado ou o pino *MR* está mantido em BAIXO. Se alguma destas duas suposições for falsa, teremos removido metade dos CIs da lista dos suspeitos! Colocamos a ponta de prova no *MR* do contador de palavras para verificar se ele vai para ALTO. Se ele está em ALTO, é provável que a linha de clock de transmissão não esteja aberta, o FF *Y* funcione, e não há razão para suspeitar do monoestável. Tudo isto pode ser verificado colocando-se a ponta de prova lógica em *MR* e injetando-se um pulso na entrada *T* do monoestável. Se detectarmos um pulso negativo em *MR* (com largura aproximada de um ciclo de clock), teremos eliminado as causas 5, 6 e 7. Um diagrama de depuração é mostrado na Fig. 9-39. À medida que você for seguindo as ramificações, note que testes sempre são feitos em pontos do circuito que eliminam tantas possibilidades de falha quantas forem possíveis. Este diagrama não cobre todas as falhas possíveis neste circuito; e, quando ele conclui que um componente está ruim, pode ser necessário alguma pesquisa para confirmar que o chip está ruim mesmo, antes de trocá-lo. Entretanto, ele demonstra a lógica que um bom pesquisador de falhas deveria utilizar para isolar o problema. Um bom pesquisador é como um detetive, sempre procurando por pistas e fazendo deduções inteligentes a partir dos fatos.

9-12 COMPARADORES DE MAGNITUDE

Um outro membro útil da categoria MSI de CIs é o *comparador de magnitude*. Ele é um circuito lógico combinacional que compara duas quantidades binárias de entrada e gera saídas para indicar qual delas tem a maior magnitude. A Fig. 9-40 mostra o símbolo lógico e a tabela-verdade para o comparador de magnitude de quatro bits 74HC85, que também está disponível como o 7485 e como o 74LS85.

Entradas de Dados

O 74HC85 compara dois números binários de quatro bits *sem sinal*. Um deles é $A_3A_2A_1A_0$, que é denominada de pa-

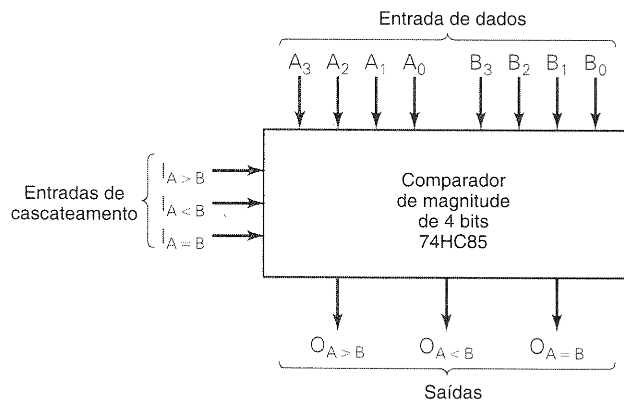


TABELA-VERDADE

ENTRADAS DE COMPARAÇÃO				ENTRADAS DE CASCATEAMENTO			SAÍDAS		
A_3, B_3	A_2, B_2	A_1, B_1	A_0, B_0	$I_{A>B}$	$I_{A<B}$	$I_{A=B}$	$O_{A>B}$	$O_{A<B}$	$O_{A=B}$
$A_3>B_3$	X	X	X	X	X	X	H	L	L
$A_3<B_3$	X	X	X	X	X	X	L	H	L
$A_3=B_3$	$A_2>B_2$	X	X	X	X	X	H	L	L
$A_3=B_3$	$A_2<B_2$	X	X	X	X	X	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1>B_1$	X	X	X	X	H	L	L
$A_3=B_3$	$A_2=B_2$	$A_1<B_1$	X	X	X	X	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0>B_0$	X	X	X	H	L	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0<B_0$	X	X	X	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	H	L	L	H	L	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	L	H	L	L	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	X	X	H	L	L	H
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	L	L	L	H	H	L
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0=B_0$	H	H	L	L	L	L

H = Nível de Tensão ALTO
L = Nível de Tensão BAIXO
X = Irrelevante

Fig. 9-40 Símbolo lógico e tabela-verdade para o comparador de magnitude de quatro bits 74HC85 (7485, 74LS85).

lavra A; o outro é $B_3B_2B_1B_0$, que é chamado de palavra B. O termo *palavra* é usado no campo dos computadores digitais para designar um grupo de bits que representam algum tipo de informação específica. Aqui, palavra A e palavra B representam quantidades numéricas.

Saídas

O 74HC85 possui três saídas ativas em nível ALTO. A saída $O_{A>B}$ estará em ALTO quando a magnitude da palavra A for maior do que a magnitude da palavra B. A saída $O_{A<B}$ estará em ALTO quando a magnitude da palavra A for menor do que a magnitude da palavra B. A saída $O_{A=B}$ estará em ALTO quando a palavra A e a palavra B forem idênticas.

Entradas de Cascateamento

As entradas de cascateamento fornecem uma maneira de expandir a operação de comparação para mais do que quatro bits, cascateando dois ou mais comparadores de quatro bits. Note que as entradas de cascateamento são identificadas do mesmo modo que as saídas. Quando uma comparação de quatro bits está sendo feita, como na Fig. 9-41(a), as entradas de cascateamento devem ser conectadas como mostrado, para produzirem as saídas corretas.

Quando dois comparadores são cascateados, as saídas do comparador de mais baixa ordem são conectadas nas entradas correspondentes do comparador de mais alta ordem. Isto é mostrado na Fig. 9-41(b), onde o comparador da esquerda está comparando os quatro bits de mais baixa ordem das duas palavras de oito bits: $A_7A_6A_5A_4A_3A_2A_1A_0$ e $B_7B_6B_5B_4B_3B_2B_1B_0$. Suas saídas são ligadas nas entradas de cascateamento do comparador da direita, que está comparando os bits de ordem mais alta. As saídas do comparador de mais alta ordem são as saídas finais que indicam o resultado da comparação de oito bits.

EXEMPLO 9-16

Descreva a operação do circuito para comparação de oito bits da Fig. 9-41(b) para os seguintes casos:

(a) $A_7A_6A_5A_4A_3A_2A_1A_0$ = 10101111; $B_7B_6B_5B_4B_3B_2B_1B_0$ = 10110001

(b) $A_7A_6A_5A_4A_3A_2A_1A_0$ = 10101111; $B_7B_6B_5B_4B_3B_2B_1B_0$ = 10101001

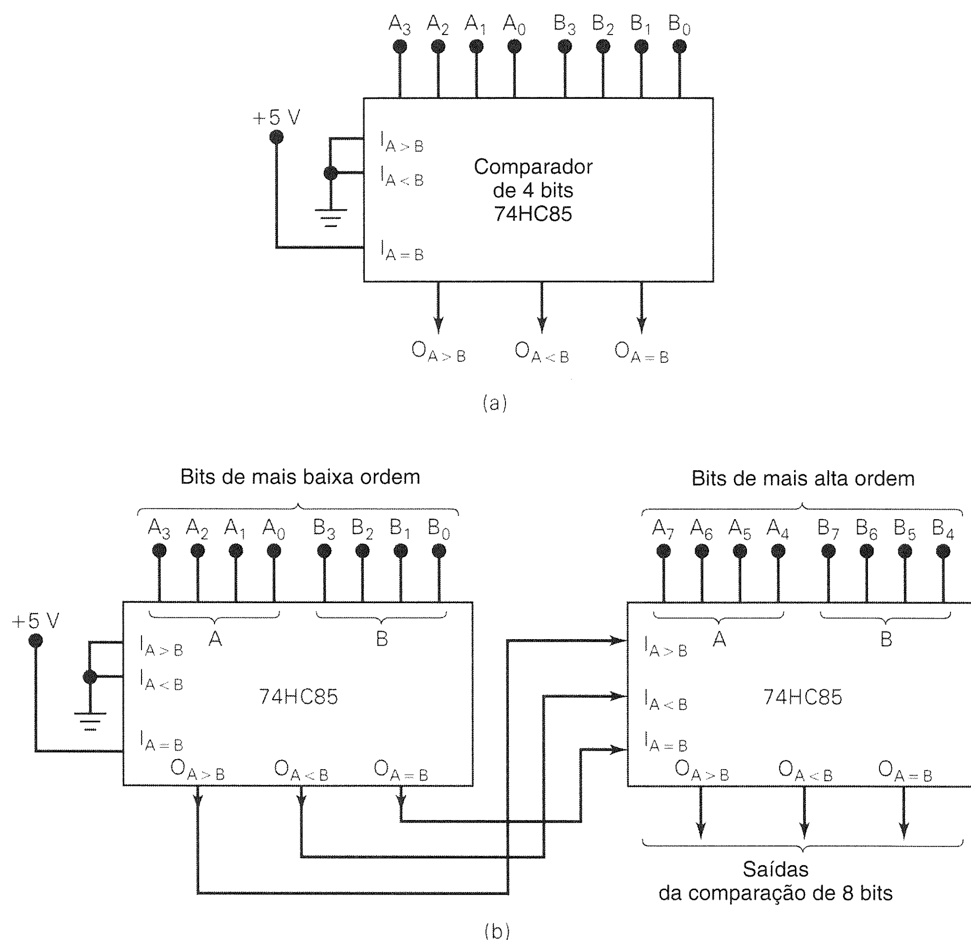


Fig. 9-41 (a) 74HC85 ligado como um comparador de quatro bits; (b) dois 74HC85s cascateados para realizar uma comparação de oito bits.

Solução

(a) O comparador de mais alta ordem compara suas entradas $A_7A_6A_5A_4$ = 1010 e $B_7B_6B_5B_4$ = 1011 e produz $O_{A<B}$= 1, não importando os níveis que estão sendo aplicados nas suas entradas de cascateamento pelo comparador de mais baixa ordem. Em outras palavras, uma vez que o comparador de mais alta ordem detecta uma diferença entre os bits de mais alta ordem das duas palavras de oito bits, ele sabe qual palavra de oito bits é maior sem ter que verificar o resultado da comparação de mais baixa ordem.

(b) O comparador de mais alta ordem detecta $A_7A_6A_5A_4$ = $B_7B_6B_5B_4$ = 1010, e portanto ele deve verificar nas suas entradas de cascateamento o resultado da comparação de mais baixa ordem. O comparador de mais baixa ordem tem $A_3A_2A_1A_0$ = 1111 e $B_3B_2B_1B_0$ = 1001, o que produz 1 para a sua saída $O_{A>B}$ e para a entrada $I_{A>B}$ do comparador de mais alta ordem. O comparador de mais alta ordem detecta este 1, e, como suas entradas de dados são iguais, ele produz um nível ALTO na saída $O_{A>B}$ para indicar o resultado da comparação de oito bits.

Aplicações

Os comparadores de magnitude também são úteis em aplicações de controle, onde um número binário que representa uma variável física sendo controlada (por exemplo: posição, velocidade ou temperatura) é comparado com um valor de referência. As saídas do comparador são usadas para atuar nos circuitos que levam a variável física em direção ao valor de referência. O exemplo a seguir ilustrará uma aplicação. Examinaremos uma outra aplicação de comparadores no Problema 9-49.

EXEMPLO 9-17

Considere um termostato digital no qual a temperatura ambiente de um quarto é convertida para um número digital e aplicada nas entradas *A* de um comparador. A temperatura desejada do quarto, informada através de um teclado, é armazenada num registrador que está conectado nas entradas *B*. Se $A < B$, o aquecedor deveria ser ativado para esquentar o quarto. O aquecedor deveria continuar ligado enquanto $A = B$ e desligar quando $A > B$. Conforme o quarto fosse esfriando, o aquecedor permaneceria desligado enquanto $A = B$ e ligaria novamente quando $A < B$. Que circuito digital poderia ser usado para interfacear um comparador de magnitude com o aquecedor, para realizar esta aplicação de controle deste termostato?

Solução

Usar a saída $O_{A<B}$ para acionar diretamente o aquecedor poderia causar seu desligamento tão logo os valores se tornassem iguais. Isto pode provocar um ciclo liga-desliga constante no aquecedor quando a temperatura real estiver muito próxima do limite entre $A < B$ e $A = B$. Utilizando-se um circuito latch NOR SET-CLEAR (consulte o Cap. 5) como mostrado na Fig. 9-42, o sistema vai operar conforme o descrito. Note que $O_{A<B}$ é conectada na entrada SET e $O_{A>B}$ é conectada na entrada CLEAR do latch. Quando a temperatura for mais quente do que a desejada, o latch fica limpo, desligando o aquecedor. Quando a temperatura for mais fria que a desejada, o latch é setado, ligando o aquecedor.

Símbolo IEEE/ANSI

A Fig. 9-43 mostra o símbolo IEEE/ANSI para o comparador 74HC85. Repare que o símbolo utiliza *P* e *Q* para representar as variáveis de entrada. Esta designação é a preferida pelo padrão IEEE/ANSI.

Questões de Revisão

1. Qual é o objetivo das entradas de cascateamento do 74HC85?
2. Quais são as saídas de um 74HC85 com as seguintes entradas: $A_3A_2A_1A_0 = B_3B_2B_1B_0$ = 1001, $I_{A>B} = I_{A<B}$ = 0 e $I_{A=B}$ = 1?

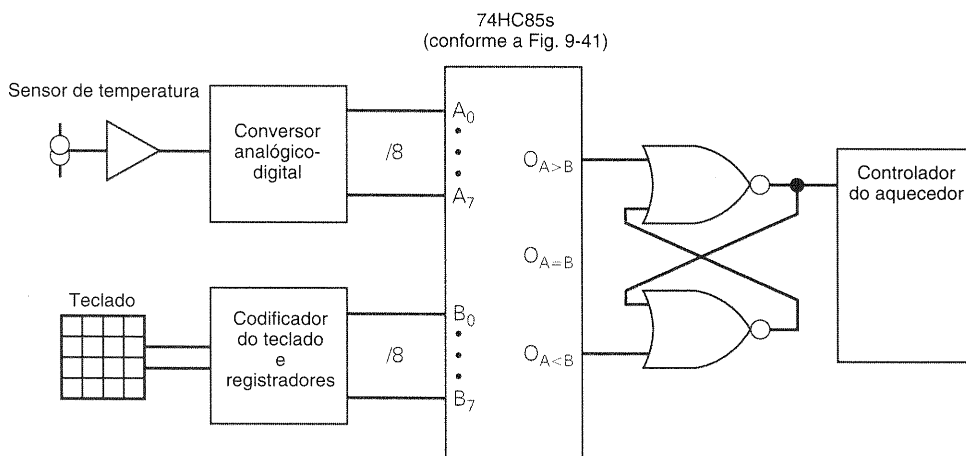


Fig. 9-42 Comparador de magnitude usado em um termostato digital.

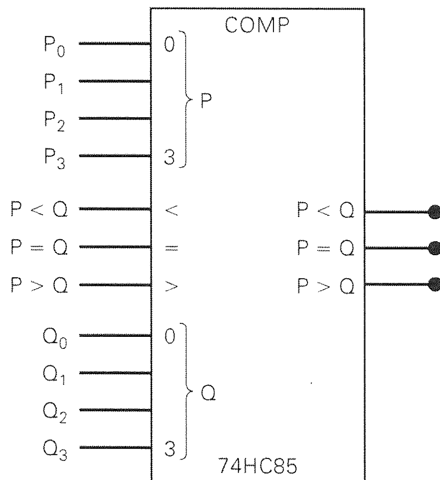


Fig. 9-43 Símbolo IEEE/ANSI para comparador.

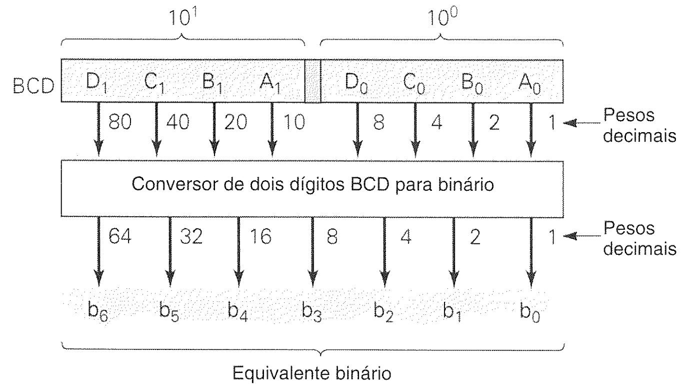


Fig. 9-44 Idéia básica para um conversor de dois dígitos BCD para binário.

9-13 CONVERSORES DE CÓDIGO

Um conversor de código é um circuito lógico que altera os dados representados em um tipo de código binário para outro tipo de código binário. O decodificador/driver BCD para 7 segmentos que apresentamos anteriormente é um conversor de código porque transforma um código de entrada BCD para o código de 7 segmentos necessário para o display a LEDs. Uma lista parcial de algumas das conversões de código mais comuns é apresentada na Tabela 9-5.

Como um exemplo de um circuito conversor de código, vamos considerar um conversor BCD para binário. Antes de analisarmos a implementação do circuito, devemos rever a representação BCD.

Valores decimais de dois dígitos variando de 00 até 99 podem ser representados em BCD por dois grupos de quatro bits. Por exemplo, 57_{10} é representado como

$$\overbrace{0101}^{5} \quad \overbrace{0111}^{7} \quad \text{(BCD)}$$

A representação binária direta para 57 decimal é

$$57_{10} = 111001_2$$

O maior valor decimal de dois dígitos é 99 que tem a seguinte representação:

$$99_{10} = 10011001 \text{ (BCD)} = 1100011_2$$

Repare que a representação binária requer apenas sete bits.

TABELA 9-5

BCD para 7 segmentos
BCD para binário
Binário para BCD
Binário para código Gray
Código Gray para binário
ASCII para EBCDIC*
EBCDIC para ASCII

*EBCDIC é um código alfanumérico desenvolvido pela IBM que é similar ao ASCII.

Idéia Básica

O diagrama da Fig. 9-44 mostra a idéia básica para um conversor de dois dígitos BCD para binário. As entradas do conversor são os dois grupos de códigos de quatro bits $D_0C_0B_0A_0$, representando 10^0 ou o dígito das unidades, e $D_1C_1B_1A_1$, representando 10^1 ou o dígito das dezenas do valor decimal. As saídas do conversor são $b_6b_5b_4b_3b_2b_1b_0$, os sete bits do binário equivalente do mesmo número decimal. Note a diferença dos pesos dos bits BCD com os pesos dos bits binários.

Uma utilização típica de um conversor BCD para binário seria onde dados em BCD de um instrumento, tal como um multímetro digital, estivessem sendo transferidos para um computador para armazenamento ou processamento. Os dados deveriam ser convertidos para binário de modo que pudessem ser operados em binário pela ULA do computador, que não teria a capacidade de realizar operações aritméticas sobre dados em BCD. A conversão BCD para binário pode ser realizada tanto por hardware quanto por software. O método de hardware (que estamos analisando no momento) geralmente é mais rápido, mas necessita de circuitos extras. O método por software não utiliza circuitos extras, mas leva mais tempo, pois o software faz a conversão passo a passo. O método escolhido para uma determinada aplicação depende de o tempo de conversão ser ou não uma característica importante.

Processo de Conversão

Os bits em uma representação BCD têm pesos decimais que são 8, 4, 2 e 1, dentro de cada grupo de código, mas que diferem por um fator de 10 de um grupo de código (dígito decimal) para o próximo. A Fig. 9-44 mostra os pesos dos bits para a representação BCD de dois dígitos.

O peso decimal de cada bit na representação BCD pode ser convertida para seu equivalente binário. Os resultados são apresentados na Tabela 9-6. Utilizando estes pesos, podemos realizar a conversão BCD para binário simplesmente fazendo o seguinte:

Calcule a soma binária dos equivalentes binários de todos os bits na representação BCD que forem 1s.

O próximo exemplo vai ilustrar.

TABELA 9-6 Equivalentes binários dos pesos decimais de cada bit BCD

Bit BCD	Peso Decimal	Equivalente Binário b_6	b_5	b_4	b_3	b_2	b_1	b_0
A_0	1	0	0	0	0	0	0	1
B_0	2	0	0	0	0	0	1	0
C_0	4	0	0	0	0	1	0	0
D_0	8	0	0	0	1	0	0	0
A_1	10	0	0	0	1	0	1	0
B_1	20	0	0	1	0	1	0	0
C_1	40	0	1	0	1	0	0	0
D_1	80	1	0	1	0	0	0	0

EXEMPLO 9-18

Converta 01010010 (BCD para o decimal 52) para binário. Repita para 10010101 (decimal 95).

Solução

Escreva os equivalentes binários de todos os 1s da representação BCD. Então some-os em binário.

```
0 1 0 1 0 0 1 0   (BCD)
  |   |     └──→ 0000010 (2 em binário)
  |   └────────→ 0001010 (10 em binário)
  └──────────→ + 0101000 (40 em binário)
                 0110100 (52 em binário)
1 0 0 1 0 1 0 1   (BCD)
|     |   |   └→ 0000001 (1 em binário)
|     |   └────→ 0000100 (4 em binário)
|     └────────→ 0001010 (10 em binário)
└────────────→ + 1010000 (80 em binário)
                 1011111 (95 em binário)
```

Implementação do Circuito

Certamente, uma maneira de implementar o circuito lógico que realiza este processo de conversão é usar circuitos somadores binários. A Fig. 9-45 mostra como dois somadores paralelos de quatro bits 74LS83s podem ser interligados para realizar a conversão. Esta é uma dentre várias configurações possíveis com somadores que funcionariam. Você pode querer rever a operação deste CI na Seção 6-14.

Os dois CIs somadores realizam a adição dos bits BCD nas combinações apropriadas de acordo com a Tabela 9-6.

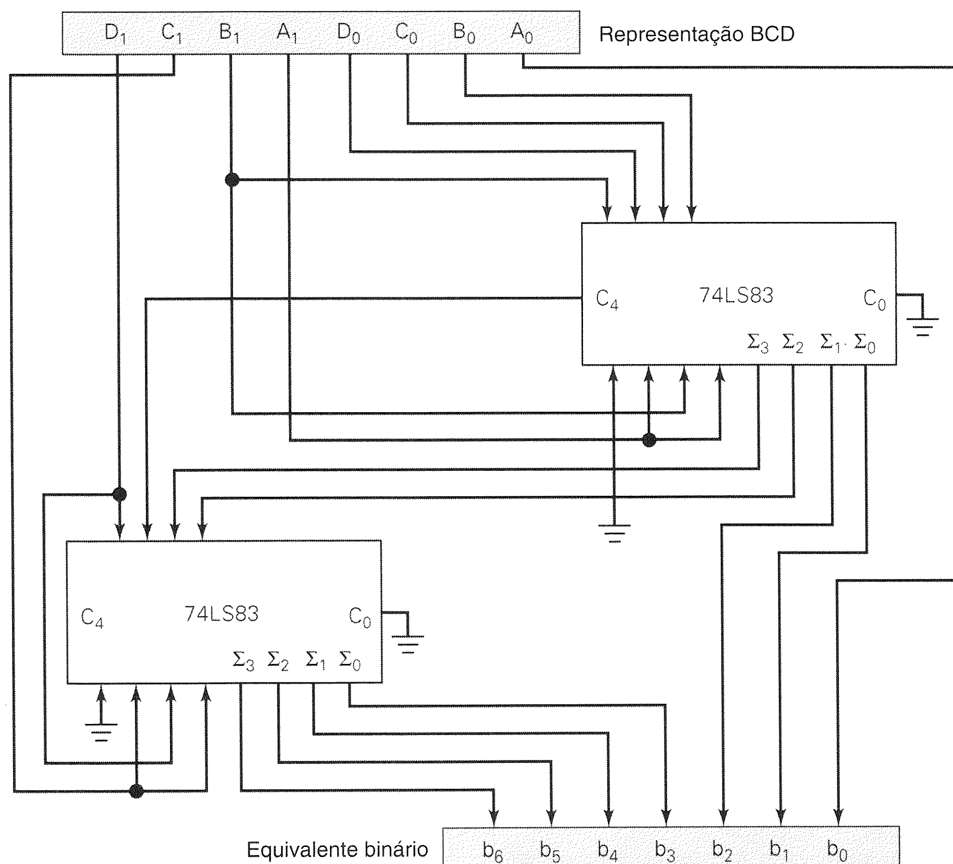


Fig. 9-45 Conversor BCD para binário implementado com somadores de quatro bits 74LS83.

Por exemplo, a Tabela 9-6 mostra que A_0 é o único bit BCD que contribui para o LSB, b_0, do equivalente binário. Já que não existe carry para esta posição, A_0 é conectado diretamente como a saída b_0. A tabela mostra também que apenas os bits BCD B_0 e A_1 contribuem para o bit b_1 da saída binária. Estes dois bits são combinados no somador superior para produzir a saída b_1. Do mesmo modo, somente os bits BCD D_0, A_1 e C_1 contribuem para o bit b_3. O somador superior combina D_0 e A_1 para gerar Σ_2, que está conectado ao somador inferior, onde C_1 é somado a ele para produzir b_3.

EXEMPLO 9-19

A representação BCD para o decimal 56 é aplicada ao conversor da Fig. 9-45. Determine as saídas Σ de cada somador e a saída binária final.

Solução

Escreva os bits da representação BCD 01010110 no diagrama do circuito. Já que $A_0 = 0$, o bit b_0 da saída é 0.

As entradas superiores do somador superior são 0011. As entradas inferiores são 0101. Este somador faz a adição para obter

$$\begin{array}{r} 0011 \\ +\ \underline{0101} \\ 1000 \end{array} = \Sigma_3\Sigma_2\Sigma_1\Sigma_0 \text{ saídas do somador superior}$$

Os bits Σ_1 e Σ_0 se tornam as saídas b_2 e b_1, respectivamente. Os bits Σ_3 e Σ_2 são ligados ao somador inferior. As entradas superiores do somador inferior são, portanto, 0010. As entradas inferiores são 0101. Este somador efetua esta adição para produzir

$$\begin{array}{r} 0010 \\ +\ \underline{0101} \\ 0111 \end{array} = \Sigma_3\Sigma_2\Sigma_1\Sigma_0 \text{ saídas do somador inferior}$$

Estes bits se tornam $b_6b_5b_4b_3$, respectivamente.

Logo, temos $b_6b_5b_4b_3b_2b_1b_0 = 0111000$ como o equivalente binário correto para o decimal 56.

Outras Implementações para Conversores de Código

Embora todos os tipos de conversores de código possam ser construídos combinando-se portas lógicas, circuitos somadores, ou outras lógicas combinacionais, os circuitos podem se tornar bastante complexos, necessitando de muitos CIs. Freqüentemente é mais eficiente usar uma memória de apenas leitura (ROM — *read-only memory*) ou um dispositivo de lógica programável (PLD — *programmable logic device*) para funcionar como um conversor de código. Como estudaremos nos Caps. 11 e 12, estes dispositivos contêm o equivalente a centenas de portas lógicas, e eles podem ser programados para fornecer uma ampla faixa de funções lógicas.

Questões de Revisão

1. O que é um conversor de códigos?
2. Quantas saídas binárias deveria ter um conversor de três dígitos BCD para binário?

9-14 BARRAMENTO DE DADOS

Na maioria dos computadores modernos, a transferência de dados se realiza através de um conjunto comum de linhas de conexão denominado **barramento de dados**. Nestes computadores organizados em barramentos, muitos dispositivos diferentes podem ter suas entradas e saídas ligadas nas linhas comuns do barramento de dados. Por isso, os dispositivos que são ligados ao barramento de dados freqüentemente possuem saídas tristate, ou são ligados ao barramento de dados através de buffers de terceiro estado.

Alguns dos dispositivos que são normalmente ligados no barramento de dados são: (1) microprocessadores, discutidos no Cap. 13; (2) chips de memórias semicondutoras, abordados no Cap. 11; e (3) conversores digitais-analógicos (DACs) e conversores analógico-digitais (ADCs), descritos no Cap. 10.

A Fig. 9-46 ilustra uma situação típica na qual um microprocessador (o chip da CPU de um microcomputador) está conectado em vários dispositivos através de um barramento de dados de 8 bits. O barramento de dados é simplesmente uma coleção de caminhos condutores sobre os quais os dados digitais são transmitidos de um dispositivo para outro. Cada dispositivo fornece uma saída de oito bits, que são enviados para as entradas do microprocessador pelas oito linhas da barra de dados. Obviamente, tendo em vista que as saídas de cada um dos três dispositivos estão conectadas nas mesmas entradas do microprocessador pelas vias do barramento de dados, devemos tomar cuidado com problemas de contenção de barramento (Seção 8-8), onde dois ou mais sinais ligados na mesma linha do barramento ficam ativos e, basicamente, competem um com o outro. Contenção de barramento é evitada se os dispositivos têm saídas tristate ou são conectados ao barramento através de buffers de terceiro estado (Seção 8-8). As entradas de habilitação de saída (*OE — output enable*) de cada dispositivo (ou de seu buffer) são usadas para garantir que apenas um dispositivo ative suas saídas em cada instante.

EXEMPLO 9-20

(a) Para a Fig. 9-46, descreva as condições necessárias para transmitir um dado do dispositivo 3 para o microprocessador.

(b) Qual é o estado do barramento de dados quando nenhum dispositivo está habilitado?

Solução

(a) HABILITAÇÃO 3 deve estar ativada; HABILITAÇÃO 1 e HABILITAÇÃO 2 devem ficar em seus estados inativos. Isto deixa os dispositivos 1 e 2 no estado de alta impedância e essencialmente desconectados do barramento. As saídas do dispositivo ficam ativadas, de modo que seus níveis lógicos aparecem nas linhas do barramento de dados e são transmitidos para as entradas do microprocessador. Podemos visualizar isto cobrindo o dispositivo 1 e o dispositivo 2 como se eles não fizessem parte do circuito; então resta apenas o

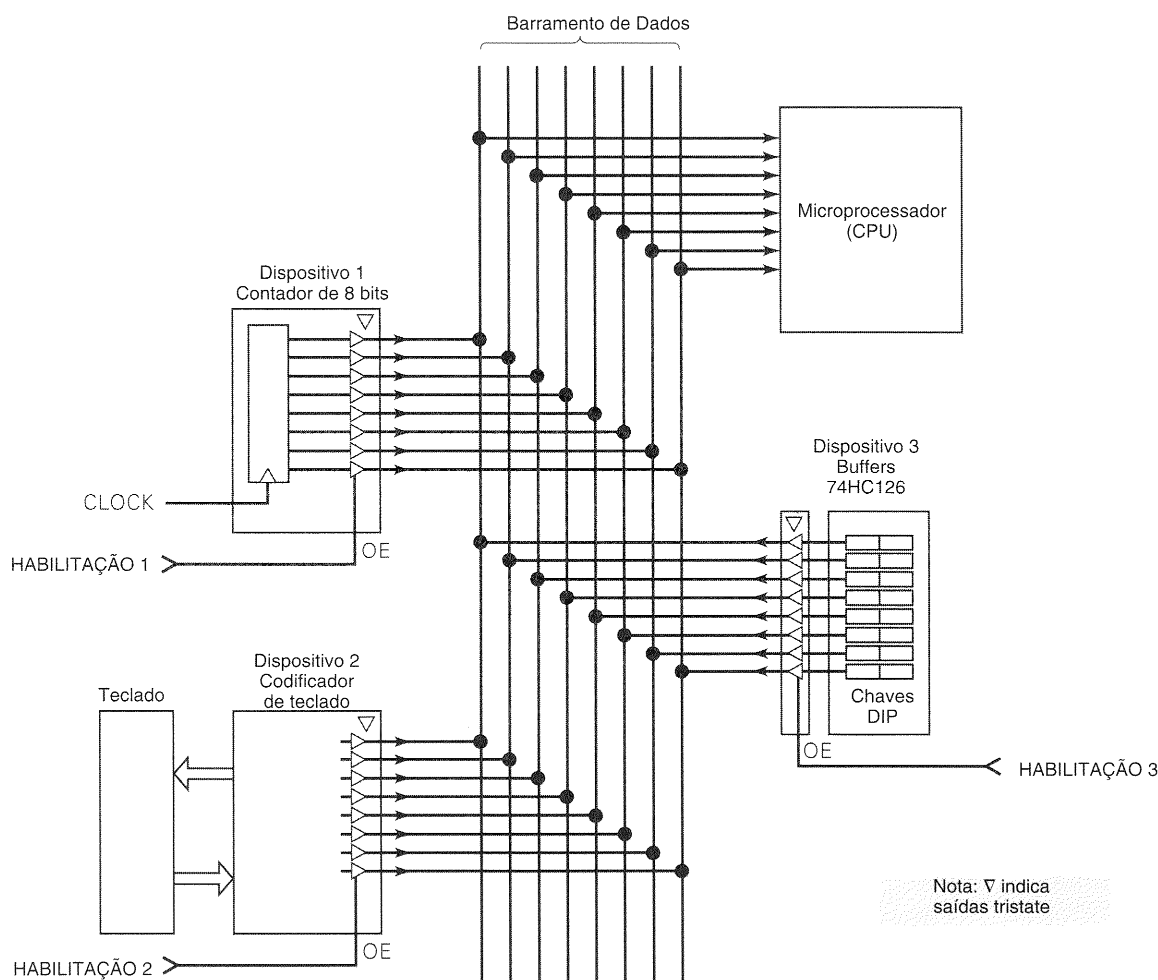


Fig. 9-46 Três dispositivos diferentes podem transmitir dados de oito bits através de um barramento de dados de oito linhas para um microprocessador; somente um dispositivo de cada vez é habilitado, de modo que a contenção de barramento é evitada.

dispositivo 3 sozinho conectado ao microprocessador através do barramento de dados.

(b) Se nenhuma das entradas de habilitação está ativada, todas as saídas dos dispositivos estão no estado de alta impedância. Isto desconecta todas as saídas dos dispositivos do barramento. Assim, não existe nível lógico definido em nenhuma das linhas da barra de dados; elas estão num estado indeterminado. Esta condição é conhecida como **barramento em flutuação**, e diz-se que cada linha do barramento está num estado de *flutuação* (indeterminado). A tela de um osciloscópio para uma linha em flutuação seria imprevisível. Uma ponta de prova lógica indicaria um nível lógico indeterminado.

Questões de Revisão

1. O que significa o termo *barramento de dados*?
2. O que é *contenção de barramento*, e o que deve ser feito para evitá-la?
3. O que é um *barramento em flutuação*?

9-15 O REGISTRADOR TRISTATE 74173/LS173/HC173

Os dispositivos conectados na barra de dados contêm registradores que mantêm os dados do dispositivo. As saídas destes registradores usualmente são conectadas em buffers tristate que permitem que eles sejam ligados no barramento de dados. Vamos demonstrar os detalhes da operação no barramento de dados usando um CI registrador que inclui buffers tristate no mesmo chip: o TTL 74173 (também disponível nas versões 74LS173 e o CMOS 74HC173). Seu diagrama lógico e a tabela-verdade são mostrados na Fig. 9-47.

O 74173 é um registrador de quatro bits com entradas e saídas paralelas. Note que as saídas dos FFs são conectadas aos buffers de terceiro estado, que fornecem as saídas O_0 até O_3. Repare também que as entradas de dados D_0 a D_3 estão conectadas nas entradas *D* dos FFs do registrador através de um circuito lógico. Esta lógica permite dois modos de operação: (1) *carga*, onde os dados nas entradas D_0 a D_3 são transferidos para os FFs na subida do pulso de clock em *CP*; e (2) *manutenção*, onde os dados no registrador não mudam quando ocorre uma transição de subida em *CP*.

Entradas					Saídas dos FFs
MR	CP	$\overline{IE_1}$	$\overline{IE_2}$	D_n	Q
H	X	X	X	X	L
L	L	X	X	X	Q_0
L	↑	H	X	X	Q_0
L	↑	X	H	X	Q_0
L	↑	L	L	L	L
L	↑	L	L	H	H

Quando $\overline{OE_1}$ ou $\overline{OE_2}$ estão em ALTO, a saída está no estado de alta impedância; entretanto, isto não afeta o conteúdo ou a operação seqüencial do registrador.

H = Nível de Tensão ALTO
L = Nível de Tensão BAIXO
X = Irrelevante

Q_0 = saída antes da transição de subida

Diagrama Lógico

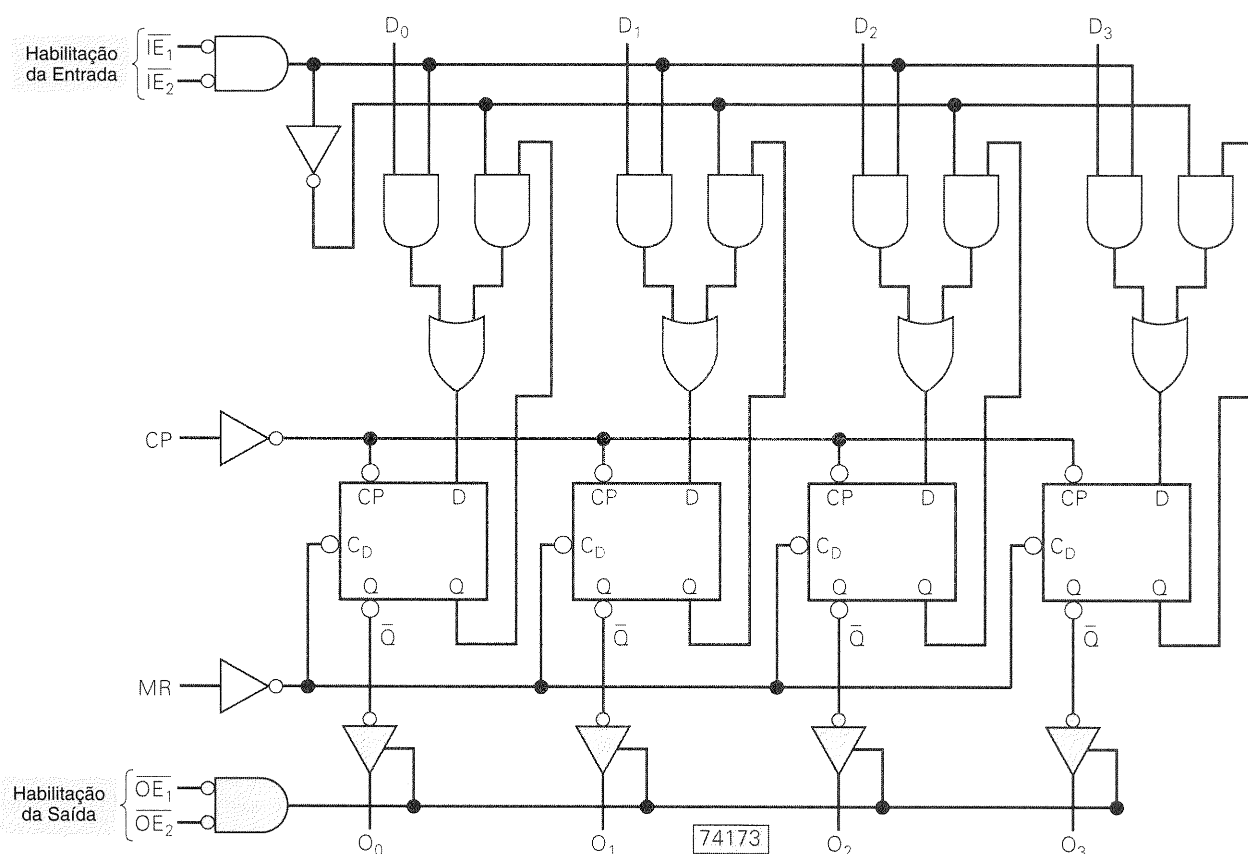


Fig. 9-47 Tabela-verdade e diagrama lógico para o registrador tristate 74173. (Cortesia da Fairchild, uma empresa da Schlumberger.)

EXEMPLO 9-21

(a) Quais são as condições de entrada que produzem a operação de carga?
(b) Quais são as condições de entrada que produzem a operação de manutenção?
(c) Que condições de entrada vão permitir que as saídas internas dos registradores apareçam nas saídas Q_0 a Q_3?

Solução

(a) As duas últimas linhas da tabela-verdade mostram que cada saída *Q* assume o valor presente na sua entrada *D* quando uma transição de subida ocorre em *CP*, desde que *MR* esteja em BAIXO e *as duas* entradas de habilitação, $\overline{IE_1}$ e $\overline{IE_2}$ estejam em BAIXO.
(b) A terceira e a quarta linha da tabela-verdade informam que, quando qualquer das entradas $\overline{IE}$ estiver em ALTO, as entradas *D* não têm efeito algum, e as saídas *Q* man-

terão seus valores correntes quando uma transição de subida ocorrer.

(c) Os buffers de saída são habilitados quando *as duas* entradas de habilitação de saída, $\overline{OE_1}$ e $\overline{OE_2}$, estiverem em BAIXO. Isto faz as saídas dos registradores passarem para as saídas externas O_0 a O_3. Se alguma das entradas de habilitação de saída ficar em ALTO, os buffers são desabilitados, e as saídas estarão em estado de alta impedância.

Note que as entradas $\overline{OE}$ não têm efeito sobre a operação de carregamento. Elas são usadas apenas para controlar se as saídas dos registradores são passadas ou não para as saídas externas.

O símbolo lógico para o 74173/LS173/HC173 é apresentado na Fig. 9-48. Incluímos a notação IEEE/ANSI "&" para indicar a relação AND dos dois pares de entradas de habilitação.

Questões de Revisão

1. Admita que as entradas *IE* estejam em BAIXO e que $D_0D_1D_2D_3$ = 1011. Que níveis lógicos estão presentes nas entradas *D* dos FFs?
2. *Verdadeiro ou falso:* O registrador não pode ser carregado quando a entrada de reset geral (*MR*) é mantida em ALTO.
3. Quais serão os níveis de saída quando *MR* = ALTO e ambas as entradas *OE* estiverem em BAIXO?

9-16 OPERAÇÃO DO BARRAMENTO DE DADOS

A barra de dados é muito importante em sistemas computacionais, e seu significado não será apreciado até nossos estudos posteriores sobre memórias e microprocessadores. No momento, ilustraremos a operação do barramento de dados para uma transferência de dados registrador para registrador. A Fig. 9-49 mostra um sistema organizado em barramento com três registradores tristate 74HC173. Note que cada registrador tem o seu par de entradas $\overline{OE}$ interligadas como uma entrada $\overline{OE}$, e analogamente para as entradas $\overline{IE}$. Note também que os registradores são referenciados como registradores *A*, *B* e *C* de cima para baixo. Isto está indicado pelo subscrito em cada entrada e saída.

Nesta configuração, o barramento de dados possui quatro linhas denominadas de DB_0 a DB_3. As saídas correspondentes de cada registrador são conectadas na mesma linha do barramento (por exemplo, O_{3A}, O_{3B} e O_{3C} são conectadas em DB_3). Como os três registradores têm suas saídas conectadas juntas, é imperativo que somente um registrador tenha suas saídas habilitadas e que as saídas dos outros dois registradores permaneçam em alta impedância. Caso contrário, haveria contenção de barramento (dois ou mais conjuntos de saídas disputando entre si), produzindo níveis incorretos no barramento e possível dano nos buffers de saída dos registradores.

As entradas correspondentes dos registradores também são ligadas na mesma linha do barramento (por exemplo, D_{3A}, D_{3B} e D_{3C} são conectadas em DB_3). Assim, os níveis no barramento estarão sempre prontos para serem transferidos para um ou mais registradores dependendo das entradas $\overline{IE}$.

Operação de Transferência de Dados

O conteúdo de qualquer dos três registradores pode ser transferido de forma paralela, pela barra de dados, para um dos outros registradores, através da aplicação adequada de níveis lógicos nas entradas de habilitação dos registradores. Num sistema típico, a unidade de controle do computador (isto é, a CPU) gera os sinais que selecionam qual registrador vai colocar seus dados no barramento de dados e qual deles vai pegar os dados do barramento. O exemplo a seguir ilustra isto.

EXEMPLO 9-22

Descreva os sinais necessários para transferir [*A*] → [*C*].

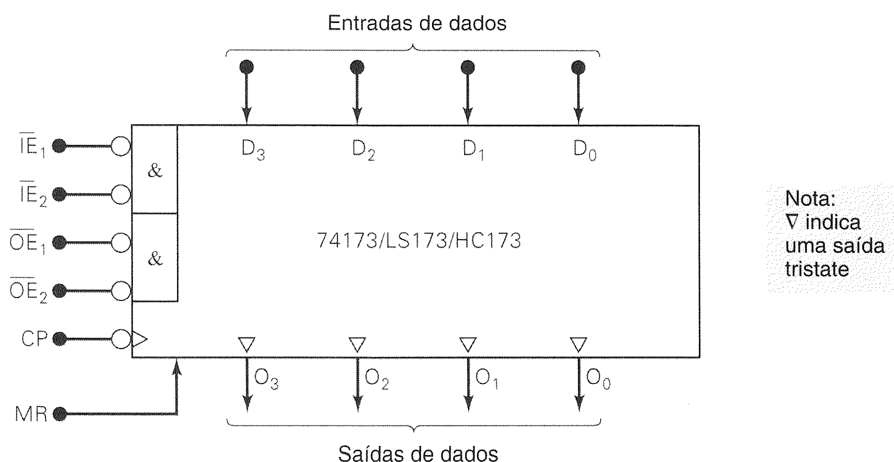


Fig. 9-48 Símbolo lógico para o CI 74173/LS173/HC173.

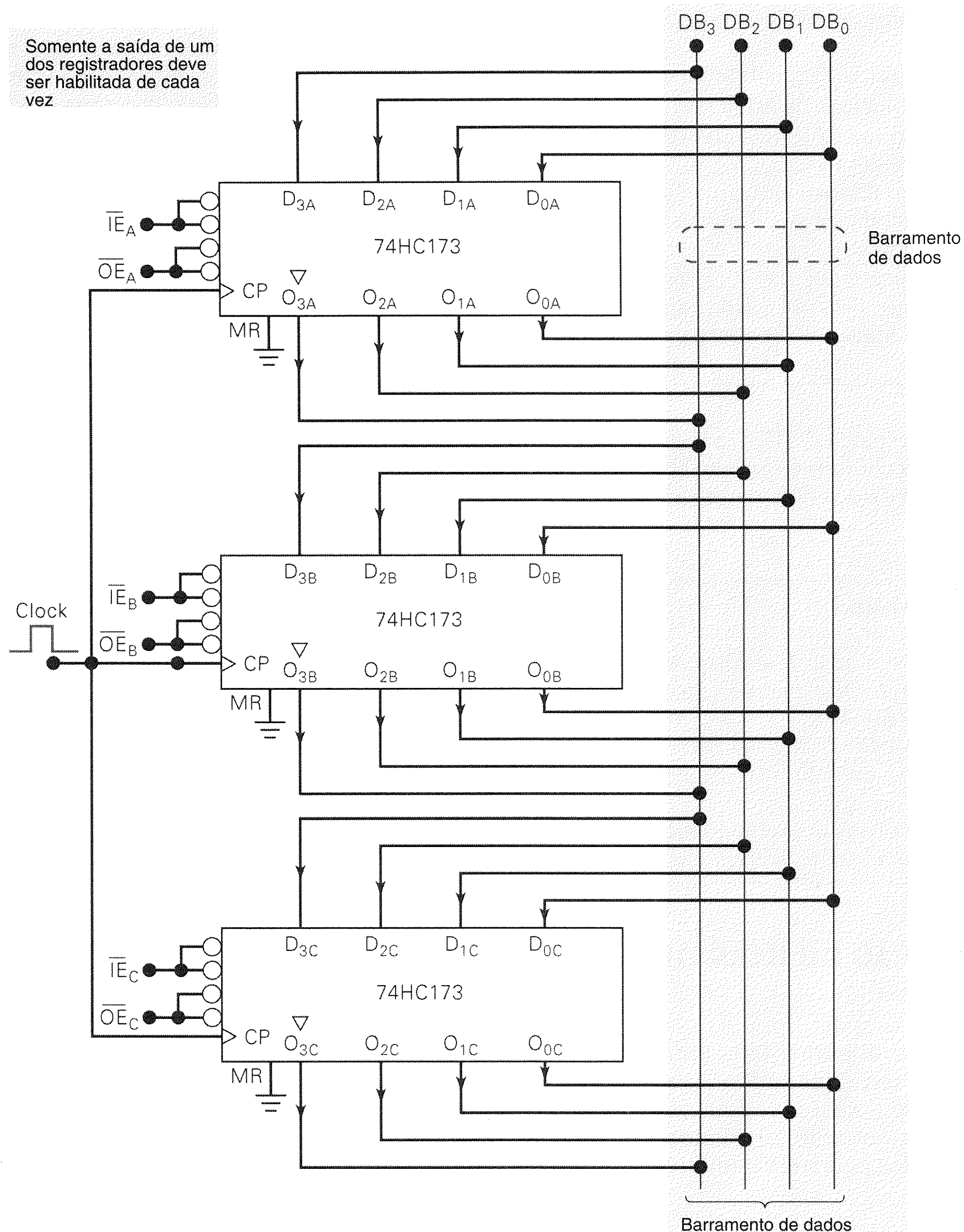


Fig. 9-49 Registradores tristate conectados num barramento de dados.

Solução

Em primeiro lugar, apenas o registrador A deve ter suas saídas habilitadas. Isto é, precisamos de

$$\overline{OE}_A = 0 \qquad \overline{OE}_B = \overline{OE}_C = 1$$

Isto coloca o conteúdo do registrador A nas linhas do barramento de dados.

A seguir, somente o registrador C deve ter suas entradas habilitadas. Para isto, devemos ter

$$\overline{IE}_C = 0 \qquad \overline{IE}_A = \overline{IE}_B = 1$$

Isto permite que apenas o registrador C aceite os dados da barra de dados quando ocorre a subida do sinal de clock.

Finalmente, um pulso de clock é necessário para realmente transferir os dados do barramento para os flip-flops do registrador C.

Sinais do Barramento

O diagrama de tempo na Fig. 9-50 mostra os diversos sinais envolvidos na transferência do dado 1011 do registrador A para o registrador C. Considere que as linhas $\overline{IE}$ e $\overline{OE}$, que não estão mostradas, estão nos seus estados inativos em ALTO. Antes de t_1, as linhas $\overline{IE}_C$ e $\overline{OE}_A$ também estão em ALTO, de modo que todas as saídas dos registradores estão desabilitadas, e nenhum dos registradores vai colocar seus

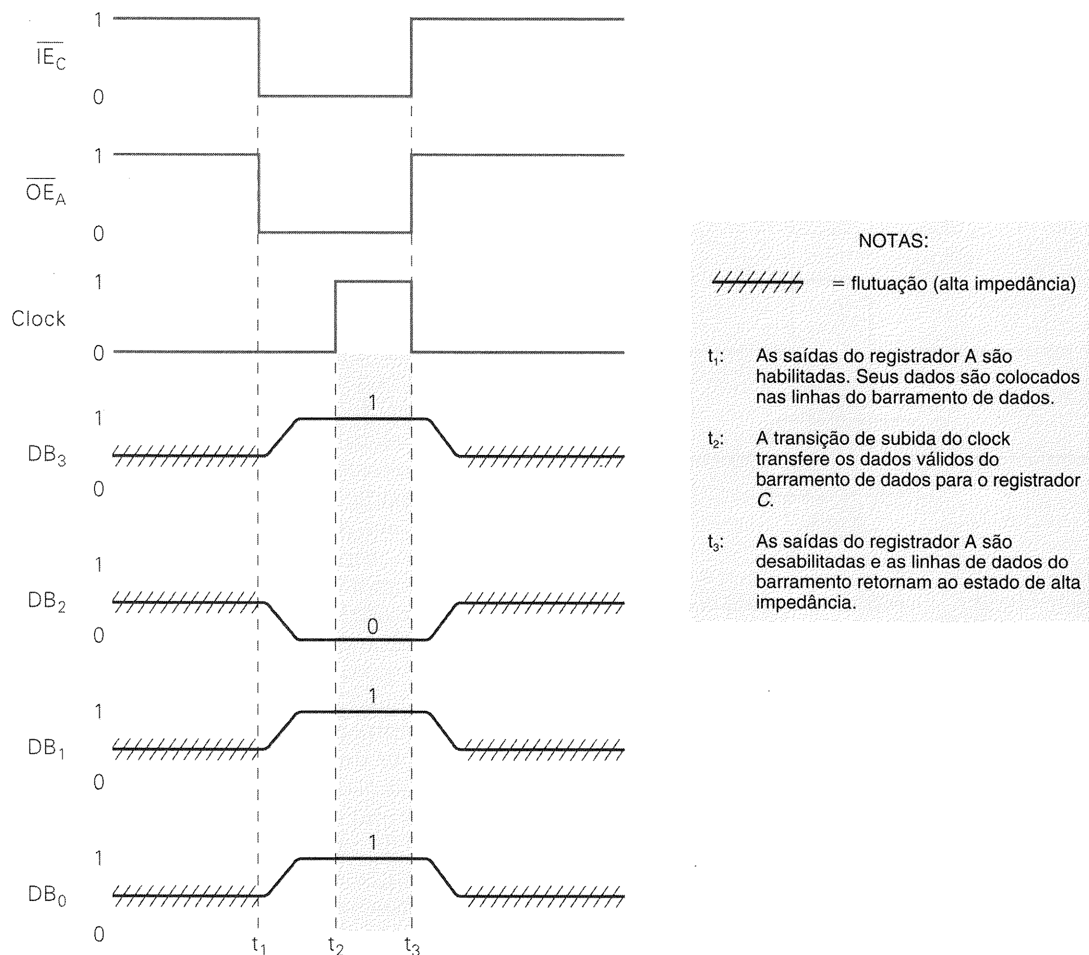


Fig. 9-50 Atividade dos sinais durante a transferência do dado 1011 do registrador *A* para o registrador *C*.

dados nas linhas do barramento. Em outras palavras, o barramento de dados está em alta impedância ou no estado de "flutuação", conforme representado pelas linhas riscadas no diagrama de tempo. O estado de alta impedância não corresponde a nenhum nível de tensão em particular.

Em t_1, as entradas $\overline{IE_C}$ e $\overline{OE_A}$ são ativadas. As saídas do registrador *A* são habilitadas, e as linhas de DB_3 a DB_0 começam a mudar de alta impedância para os níveis lógicos 1011. Após permitir o tempo para que os níveis lógicos se estabilizem no barramento, a transição de subida do clock é aplicada em t_2. Esta transição de subida transfere estes níveis lógicos para o registrador *C*, pois $\overline{IE_C}$ está ativo. Se a subida ocorre antes que o barramento assuma níveis lógicos válidos, dados imprevisíveis serão transferidos para *C*.

Em t_3, as linhas $\overline{IE_C}$ e $\overline{OE_A}$ retornam aos seus estados inativos. Em resultado, as saídas do registrador *A* vão para o estado de alta impedância. Isto remove os dados de saída do registrador *A* do barramento, e as linhas do barramento retornam ao estado de alta impedância.

Repare que as linhas de dados do barramento apresentam níveis lógicos válidos somente durante o intervalo de tempo no qual as saídas do registrador *A* estão habilitadas. Em todos os outros instantes, as linhas da barra de dados estão em flutuação, e não existe maneira fácil de prever o que apareceria se fossem mostradas na tela de um osciloscópio. Uma ponta de prova lógica daria a indicação de "indeterminado" se estivesse monitorando uma linha de barramento em flutuação. Note também a taxa relativamente lenta com que os sinais nas linhas do barramento estão mudando. Embora este efeito tenha sido exagerado no diagrama, ele é uma característica comum de sistemas em barramentos e é causado pela carga capacitiva de cada linha. Esta carga consiste na combinação das capacitâncias parasitas e das capacitâncias de cada entrada e saída conectada na linha.

Diagrama de Tempo Simplificado do Barramento

O diagrama de tempo na Fig. 9-50 mostra os sinais em cada uma das quatro linhas de dados do barramento. Este mesmo tipo de atividade de sinais ocorre em sistemas digitais que utilizam barra de dados mais comuns com 8, 16 ou 32 linhas. Para estes barramentos mais largos, os diagramas de tempo como os da Fig. 9-50 seriam excessivamente grandes e complicados. Existe um método simplificado para mostrar a atividade dos sinais, que ocorre num conjunto de linhas do barramento que utiliza somente uma única forma de onda temporal para representar o conjunto completo de linhas do barramento. Isto é ilustrado na Fig. 9-51 para a mesma situação de transferência de dados apresentada na

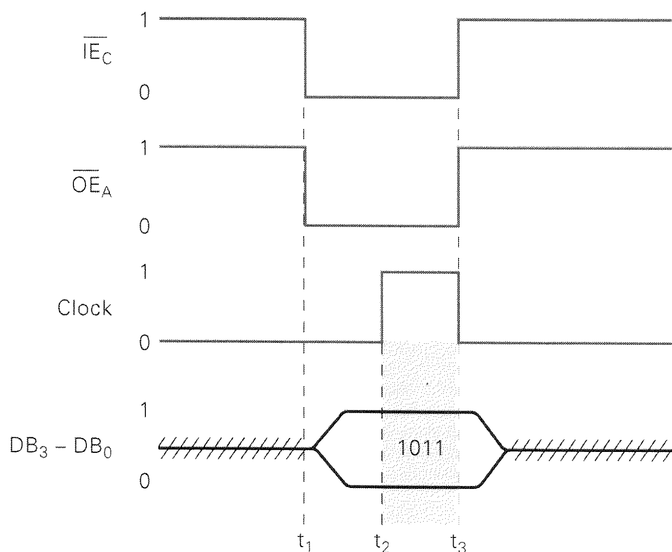


Fig. 9-51 Modo simplificado de mostrar a atividade dos sinais nas linhas do barramento de dados.

Fig. 9-50. Note como a atividade do barramento de dados está representada. Note, especialmente, como o dado válido 1011 está indicado no diagrama durante o intervalo t_2-t_3. Daqui por diante, geralmente utilizaremos este diagrama de tempo simplificado para barramentos.

Expandindo o Barramento

A operação de transferência de dados no barramento de quatro linhas, da Fig. 9-49, é típica dos barramentos de maior largura encontrados na maioria dos computadores e em outros sistemas digitais, usualmente com barramentos de 8, 16 ou 32 linhas de dados. Estes barramentos de maior largura, geralmente, têm mais do que três dispositivos conectados na barra, mas a operação básica de transferência de dados é a mesma: *um dispositivo tem suas saídas habilitadas, de modo que seus dados são colocados no barramento; um outro dispositivo tem suas entradas habilitadas, de modo que pode pegar estes dados do barramento e armazená-los em seus circuitos internos na borda apropriada do clock.*

O número de linhas do barramento de dados depende do tamanho da **palavra** de dados (unidade dos dados) que é transferida pelo barramento. Um computador que tem um tamanho de palavra de 8 bits vai ter um barramento de dados de 8 bits, um computador que tem um tamanho de palavra de 16 bits vai ter um barramento de dados de 16 bits e assim por diante. O número de dispositivos conectados no barramento de dados varia de um computador para outro e depende de fatores tais como a quantidade de memória do computador e o número de dispositivos de entrada e saída que devem se comunicar com a CPU através do barramento de dados.

Todas as saídas de dispositivos devem ser conectadas ao barramento através de buffers de terceiro estado. Alguns dispositivos, como o registrador 74173, possuem estes buffers no mesmo chip. Outros dispositivos vão precisar ser conectados ao barramento por um CI chamado **driver de barramento**. Um CI driver de barramento tem saídas de terceiro estado com impedância muito baixa, que podem carregar e descarregar rapidamente a capacitância do barramento. Esta capacitância do barramento representa o efeito cumulativo de todas as capacitâncias parasitas das diferentes entradas e saídas ligadas no barramento, e pode causar deterioração dos tempos de transição dos sinais do barramento, se não forem acionadas por uma fonte de sinal de baixa impedância. A Fig. 9-52 mostra um CI driver de barramento octal 74HC541 conectando as saídas de um conversor analógico-digital (A/D)

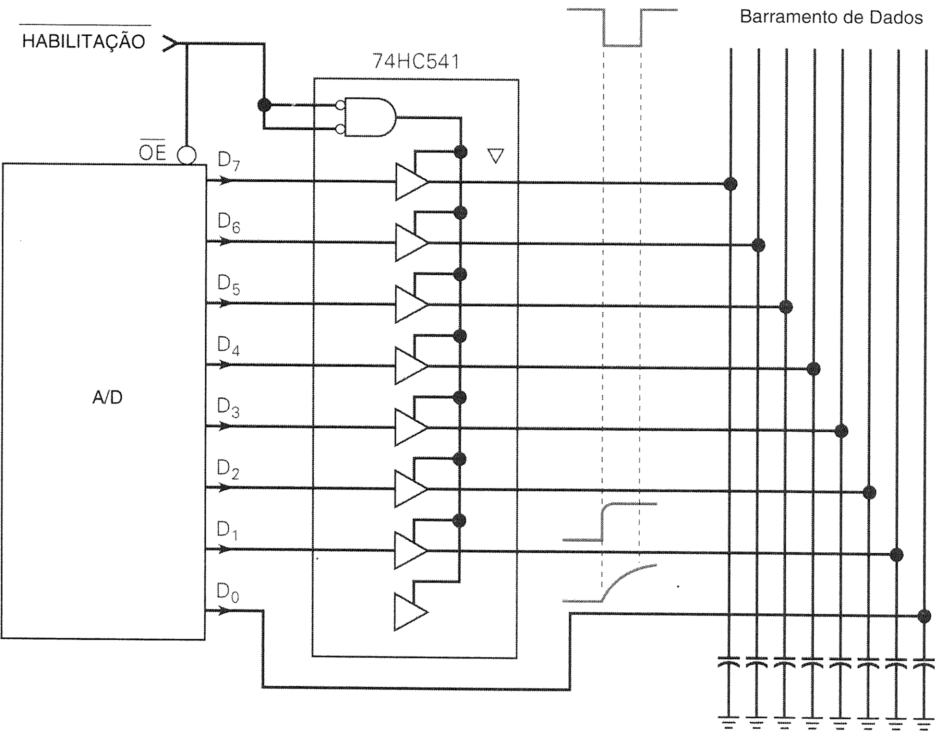


Fig. 9-52 Um driver de barramento octal 74HC541 conecta as saídas de um conversor analógico-digital (A/D) num barramento de dados de oito linhas. A saída D_0 está conectada diretamente no barramento, mostrando os efeitos capacitivos.

de 8 bits no barramento de dados. O A/D tem saídas de terceiro estado, mas falta a capacidade de acionamento para carregar a capacitância do barramento (mostrada no desenho como capacitores para a terra). Note que o bit de dados 0 está acionando o barramento diretamente sem o driver. Se o tempo de transição for muito longo, a tensão pode não alcançar o nível lógico ALTO no intervalo de tempo da habilitação. Os dois habilitadores do driver de barramento estão conectados juntos, de modo que um nível BAIXO na linha de habilitação comum permite que as saídas do A/D passem pelos buffers e daí para o barramento de dados de onde podem ser transferidas para outro dispositivo.

Representação Simplificada de Barramentos

Usualmente, muitos dispositivos estão conectados no mesmo barramento de dados. No diagrama esquemático de um circuito, isto pode produzir um arranjo confuso de linhas e conexões. Por esta razão, freqüentemente é usada uma representação mais simplificada para as conexões do barramento em diagramas de blocos e em alguns diagramas esquemáticos de circuitos. Um tipo de representação simplificada está mostrado na Fig. 9-53 para um barramento de 8 linhas.

As conexões de e para o barramento são representadas por setas largas. Os números entre colchetes indicam o número de bits que cada registrador contém, assim como o número de linhas que conectam as entradas e saídas do registrador ao barramento.

Um outro método comum para representar barramentos num diagrama esquemático é ilustrado na Fig. 9-54 para um barramento de 8 linhas. Ela mostra as oito linhas individuais de saída de um driver de barramento 74HC541, identificadas como D_7-D_0, reunidas (não conectadas) e representadas como uma única linha. Estas linhas de dados de saída reunidas são então conectadas no barramento, que também

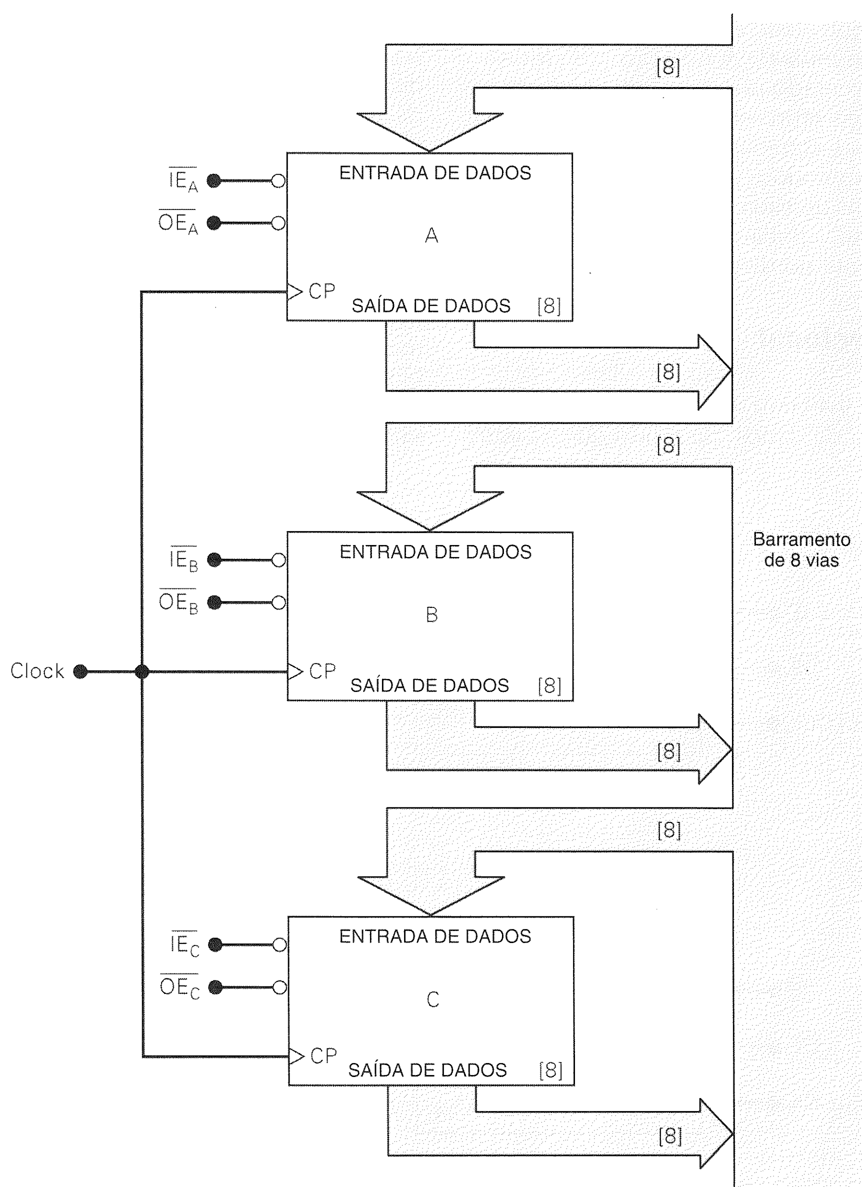


Fig. 9-53 Representação simplificada das conexões de um barramento.

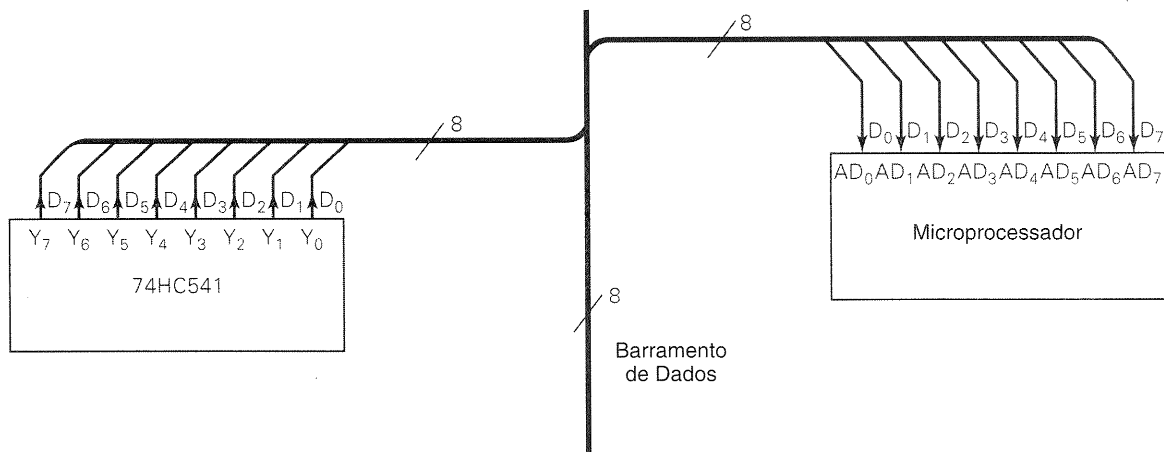


Fig. 9-54 Método de reunião das linhas para a representação simplificada de conexões do barramento de dados. O "/8" indica um barramento de dados de oito linhas.

é mostrado como uma linha (isto é, as oito linhas de dados do barramento estão reunidas). A notação "/8" indica o número de linhas representado por cada linha mais grossa. Este método é usado para representar as conexões do barramento de dados para as entradas de dados do microprocessador. Quando este método é utilizado, é muito importante identificar as pontas de cada conexão que pertencem à linha mais grossa, já que a linha da conexão não pode ser acompanhada visualmente no diagrama.

Barramento Bidirecional

Cada registrador na Fig. 9-49 tem suas entradas e saídas conectadas no barramento de dados, de modo que as entradas e saídas correspondentes estão juntas. Por exemplo, cada registrador tem sua saída O_2 ligada na entrada I_2, por causa de sua conexão comum a DB_2. Isto, naturalmente, não seria verdade se drivers de barramento externos estivessem conectados entre as saídas dos registradores e o barramento de dados.

Tendo em vista que entradas e saídas freqüentemente são conectadas juntas em sistemas de barramento, os fabricantes de CIs desenvolveram CIs que conectam suas entradas e saídas *internamente* ao chip, de modo a reduzir o número de pinos do CI e o número de conexões para o barramento. A Fig. 9-55 ilustra isto para um registrador de quatro bits. As linhas de dados de entrada (D_0 a D_3) e de saída (O_0 a O_3) separadas foram substituídas pelas linhas de entrada/saída (I/O_0 a I/O_3).

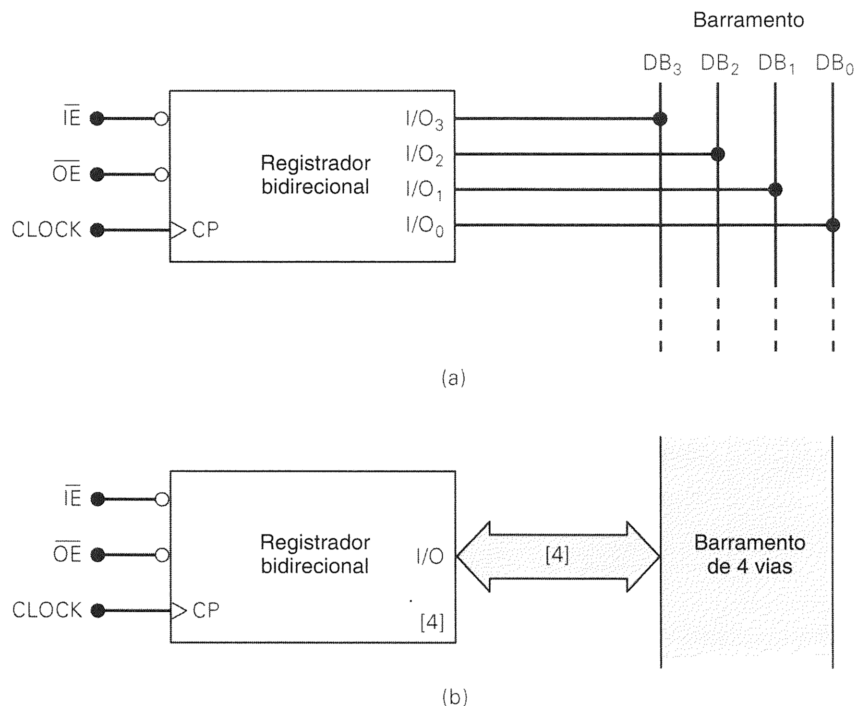


Fig. 9-55 Registrador bidirecional conectado ao barramento de dados.

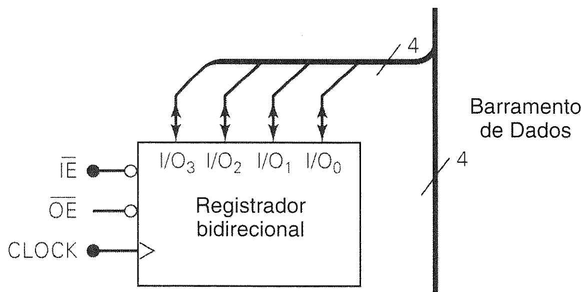


Fig. 9-56

Cada linha I/O funciona como entrada e como saída, dependendo do estado das entradas de habilitação. Assim, elas são chamadas de **linhas de dados bidirecionais**. O 74LS299 é um registrador de oito bits com linhas de I/O comuns. Muitos CIs de memória e microprocessadores têm transferência de dados bidirecional.

Retornaremos ao importante tópico do barramento de dados na nossa abordagem detalhada de sistemas de memória no Cap. 11 e em nossa introdução aos microprocessadores e microcomputadores no Cap. 12.

Questões de Revisão

1. O que acontece se $\overline{OE}_A = \overline{OE}_F$ = BAIXO na Fig. 9-49?
2. Que nível lógico fica numa linha do barramento de dados quando todos os dispositivos ligados ao barramento estão desabilitados?
3. Qual é a função de um driver de barramento?
4. Quais são os motivos para a existência de registradores com linhas de I/O comuns?
5. Desenhe novamente a Fig. 9-55(a) utilizando o método de representação com linha mais grossa. (A resposta está mostrada na Fig. 9-56.)

RESUMO

1. Um decodificador é um dispositivo cuja saída é ativada somente quando uma combinação binária única (código) está presente nas suas entradas. Muitos decodificadores MSI têm várias saídas, cada uma correspondendo a apenas uma das muitas combinações de entrada possíveis.
2. Sistemas digitais freqüentemente necessitam mostrar números decimais. Isto é feito utilizando-se displays de 7 segmentos que são acionados por chips especiais que decodificam o número binário e o traduzem para o padrão de segmentos, que representa o número para as pessoas. Os elementos dos segmentos podem ser diodos emissores de luz, cristal líquido ou eletrodos brilhantes imersos em gás neon.
3. Um codificador é um dispositivo que gera um código binário único em resposta à ativação de cada entrada individual.
4. A pesquisa de falhas em sistemas digitais envolve *observação/análise* para identificar as possíveis causas e um processo de eliminação chamado *dividir e conquistar* para isolar e identificar a causa.
5. Multiplexadores funcionam como chaves controladas digitalmente que selecionam e conectam uma entrada lógica de cada vez no pino de saída. Alternando-se, muitos sinais podem compartilhar um mesmo caminho para dados usando multiplexadores. Demultiplexadores são utilizados no fim do caminho de dados para separar os sinais que estão compartilhando o caminho e distribuí-los para seus respectivos destinos.
6. Os comparadores de magnitude servem como indicadores da relação entre dois números binários, com saídas que mostram >, < e =.
7. Freqüentemente é necessário traduzir entre os vários métodos de representar quantidades com números binários. Os conversores de código são dispositivos que recebem uma forma de representação binária e a transformam em outra.
8. Em sistemas digitais, freqüentemente muitos dispositivos devem compartilhar um mesmo caminho para os dados. Este caminho usualmente é chamado de *barramento de dados*. Embora muitos dispositivos possam acionar o barramento, pode existir apenas um "driver" de barramento de cada vez. Isto significa que os dispositivos devem se alternar para aplicar seus sinais no barramento de dados.
9. Para se alternarem, os dispositivos devem ter *saídas tristate* que podem ser desabilitadas quando um outro dispositivo estiver acionando o barramento. No estado desabilitado, a saída do dispositivo é, essencialmente, desconectada eletricamente do barramento, indo para um estado que oferece uma alta impedância tanto para a terra quanto para a fonte positiva. Dispositivos que são projetados para interfacear com um barramento têm saídas que podem estar em ALTO, BAIXO ou desabilitadas (alta impedância).

TERMOS IMPORTANTES

decodificador
decodificador/driver BCD para decimal
decodificador/driver BCD para 7 segmentos
anodo comum
catodo comum
LCD
backplane
pixel
display a gás
codificador
codificador de prioridade
observação/análise
dividir e conquistar
multiplexador (MUX)
multiplexação
conversão paralelo-série
demultiplexador (DEMUX)
barramento de dados
barramento em flutuação
palavra
driver de barramento
linhas de dados bidirecionais

PROBLEMAS

SEÇÃO 9-1

9-1. Consulte a Fig. 9-3. Determine os níveis em cada saída do decodificador para os seguintes conjuntos de condições de entrada:

(a) Todas as entradas em BAIXO.
(b) Todas as entradas em BAIXO, exceto que E_3 = ALTO.
(c) Todas as entradas em ALTO, exceto que $\overline{E}_1 = \overline{E}_2$ = BAIXO.
(d) Todas as entradas em ALTO.

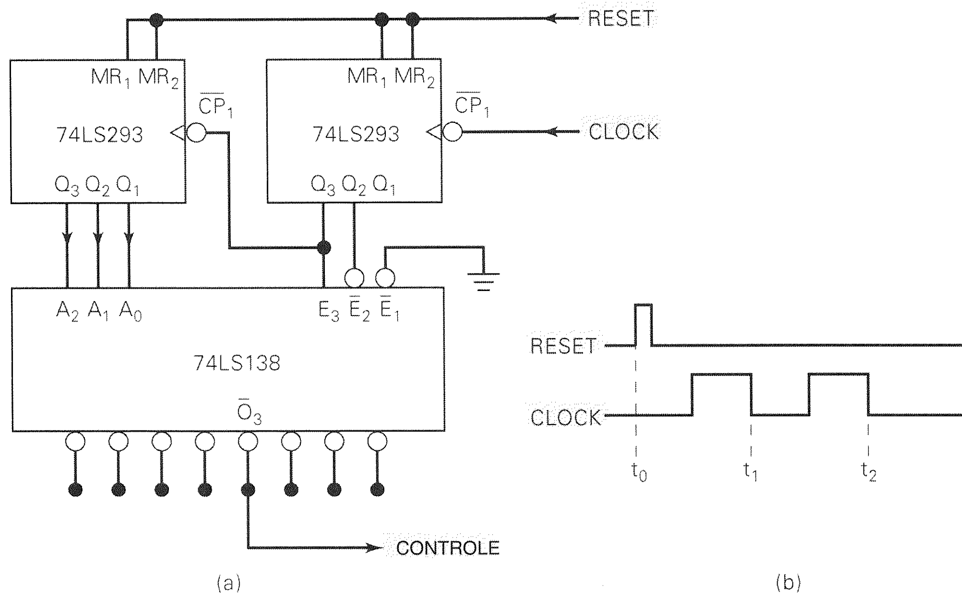


Fig. 9-57 Problemas 9-5 e 9-6.

9-2. Quantas entradas e saídas possui um decodificador que aceita 64 combinações diferentes de entrada?

9-3. Que conjunto de condições de entrada produz um nível BAIXO em $\overline{O_6}$ de um 74LS138?

D

9-4. Mostre como usar 74LS138s para formar um decodificador 4 para 16.

9-5. A Fig. 9-57 mostra como um decodificador pode ser usado na geração de sinais de controle. Admita que um pulso de RESET ocorreu no instante t_0 e determine a forma de onda de CONTROLE para 32 pulsos de clock.

D

9-6. Modifique o circuito da Fig. 9-57 para gerar uma forma de onda de CONTROLE que vai para BAIXO desde t_{20} até t_{24}. (*Sugestão:* A modificação não requer lógica adicional.)

9-7. O decodificador 7442 da Fig. 9-5 não tem uma entrada de HABILITAÇÃO. Entretanto, podemos operá-lo como um decodificador 1 de 8, não utilizando as saídas $\overline{O_8}$ e $\overline{O_9}$ e usando a entrada D como uma HABILITAÇÃO. Isto é ilustrado na Fig. 9-58. Descreva como esta configuração funciona como um decodificador 3 para 8 com habilitação e explique como o nível em D habilita ou desabilita as saídas.

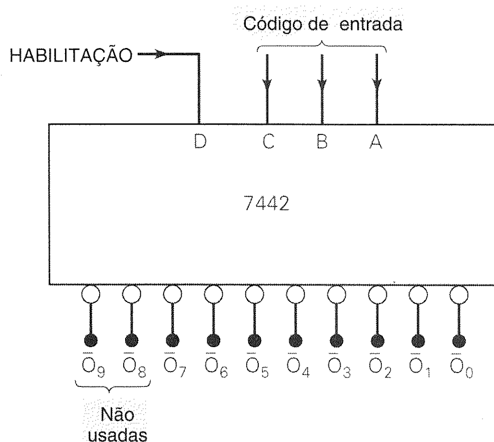


Fig. 9-58 Problema 9-7.

9-8. Considere as formas de onda na Fig. 9-59. Aplique estes sinais ao 74LS138 como segue:

$$A \rightarrow A_0 \qquad B \rightarrow A_1 \qquad C \rightarrow A_2 \qquad D \rightarrow E_3$$

Admita que $\overline{E_1}$ e $\overline{E_2}$ estão em BAIXO e desenhe as formas de onda para as saídas $\overline{O_0}$, $\overline{O_3}$, $\overline{O_6}$ e $\overline{O_7}$.

D

9-9. Modifique o circuito da Fig. 9-6, de modo que o relé K_1 fique energizado de t_3 a t_5 e K_2 fique energizado de t_6 a t_9. (*Sugestão:* Esta modificação não requer circuitos adicionais.)

SEÇÕES 9-2 E 9-3

9-10. Mostre como conectar decodificadores/drivers BCD para 7 segmentos e displays a LEDs de 7 segmentos, no circuito de relógio da Fig. 7-45, para mostrar minutos e horas. Admita que cada segmento opera com aproximadamente 10 mA a 2,5 V.

9-11. **(a)** Consulte a Fig. 9-10 e desenhe as formas de onda do segmento e do backplane em relação à terra para CONTROLE = 0. Depois desenhe a forma de onda da tensão do segmento em relação à tensão do backplane.
(b) Repita a parte (a) para CONTROLE = 1.

C, D

9-12. O decodificador/driver BCD para 7 segmentos da Fig. 9-8 contém a lógica para ativar cada segmento para as entradas BCD apropriadas. Projete a lógica para ativar o segmento *g*.

SEÇÃO 9-4

9-13. QUESTÃO DE FIXAÇÃO

Para cada sentença, indique se ela está se referindo a um decodificador ou a um codificador.

(a) Tem mais entradas do que saídas.
(b) É usado para converter acionamentos de teclas para um código binário.
(c) Somente uma saída pode ser ativada de cada vez.
(d) Pode ser usado para interfacear uma entrada BCD com um display a LEDs.
(e) Freqüentemente possui saídas do tipo driver para suportar valores maiores de *I* e *V*.

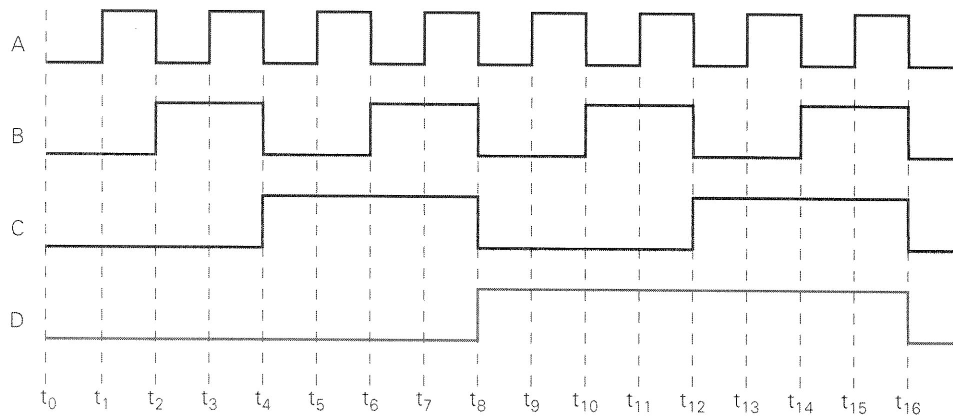


Fig. 9-59 Problemas 9-8 e 9-39.

9-14. Determine os níveis de saída para o codificador 74147 quando $\overline{A_8} = \overline{A_4} = 0$ e todas as outras entradas estão em ALTO.

9-15. Aplique os sinais da Fig. 9-59 nas entradas de um 74147 como se segue:

$$A \rightarrow \overline{A_7} \quad B \rightarrow \overline{A_4} \quad C \rightarrow \overline{A_2} \quad D \rightarrow \overline{A_1}$$

Desenhe as formas de onda das saídas do codificador.

C, D

9-16. A Fig. 9-60 mostra o diagrama de blocos de um circuito lógico usado para controlar o número de cópias feitas por uma máquina copiadora. O operador da máquina seleciona o número desejado de cópias fechando uma das chaves seletoras S_1 a S_9. Este número é codificado em BCD pelo codificador e é enviado ao circuito comparador. O operador pressiona a chave de contato momentâneo de INÍCIO, que limpa o contador e inicia a saída OPERAÇÃO em ALTO, que por sua vez é enviada para a máquina para que ela comece a fazer cópias. À medida que a máquina faz cada cópia, um pulso é gerado e levado para o contador BCD. As saídas do contador são continuamente comparadas com as saídas do codificador das chaves pelo comparador. Quando os dois números BCD são iguais, indicando que o número desejado de cópias foi feito, a saída $\overline{X}$ do comparador vai para BAIXO; isto provoca o retorno do nível de OPERAÇÃO para BAIXO e pára a máquina, de modo que nenhuma outra cópia é tirada. A ativação da chave de INÍCIO causa a repetição deste processo. Projete o circuito lógico completo para as seções de comparação e controle deste sistema.

C, D

9-17. O circuito de teclado da Fig. 9-16 foi projetado para aceitar um número decimal de três dígitos. O que aconteceria

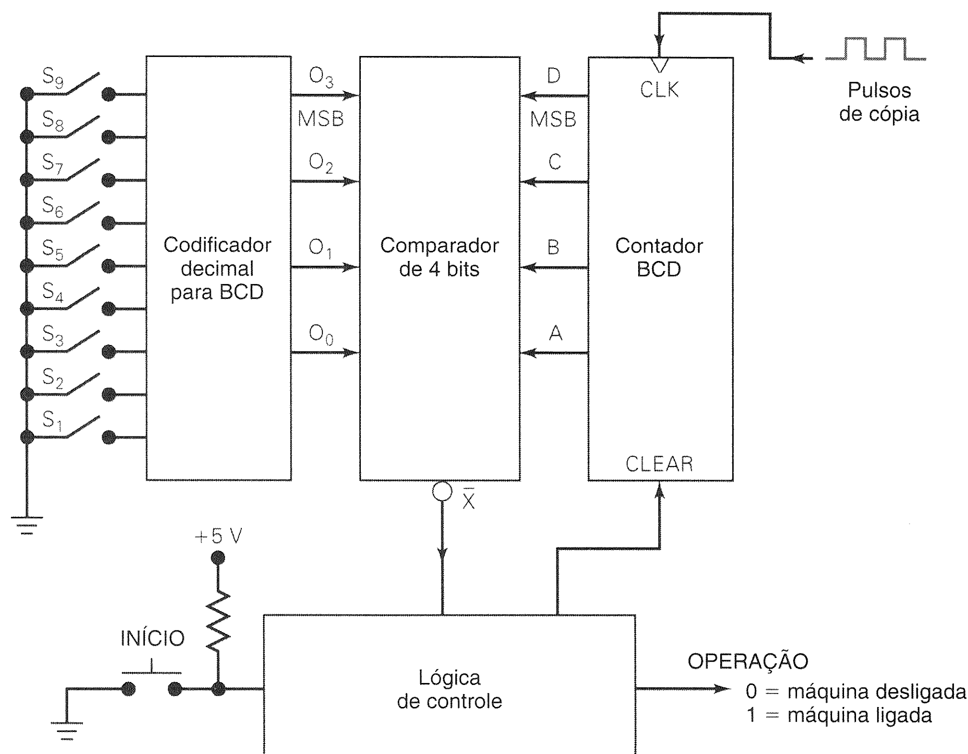


Fig. 9-60 Problemas 9-16 e 9-49.

se *quatro* teclas fossem ativadas (por exemplo, 3095)? Projete a lógica necessária a ser adicionada neste circuito, de modo que, após três dígitos terem sido acionados, qualquer outro dígito será ignorado até que a tecla LIMPA seja pressionada. Em outras palavras, se 3095 fosse digitado no teclado, os registradores de saída apresentariam 309 e iriam ignorar o 5 e qualquer outro dígito a seguir até que o circuito fosse reiniciado com LIMPA.

SEÇÃO 9-6

T

9-18. Um estudante monta o circuito de teclado da Fig. 9-16 e testa sua operação tentando entrar com uma série de números de três dígitos. Ele detecta que algumas vezes o dígito "0" entra em vez do dígito que ele pressionou. Ele também observa que isto acontece com todas as teclas de modo mais ou menos aleatório, embora seja pior para algumas teclas do que para outras. Ele substitui todos os CIs, e os problemas persistem. Quais das seguintes falhas de circuito poderiam explicar suas observações? Explique cada escolha.

(a) O estudante esqueceu de aterrar as entradas não-utilizadas da porta OR.

(b) Ele utilizou erradamente $\overline{Q}$ do monoestável em vez de Q.

(c) O efeito de trepidação de contato das chaves das teclas dura mais do que 20 ms.

(d) As saídas Y e Z estão em curto-circuito.

T

9-19. Repita o Problema 9-18 com o seguinte sintoma: os registradores e displays ficam com 0, não importando quantas vezes uma tecla seja pressionada.

T

9-20. Durante os testes do circuito da Fig. 9-16, um estudante constatou que todas as teclas ímpares resultam na entrada correta do dígito, mas todas as teclas pares resultam no número errado, do seguinte modo: a tecla "0" causa a entrada do "1", a "2" causa a entrada do "3", a "4" causa a entrada do "5" e assim por diante. Considere cada uma das seguintes falhas como possíveis causas do mau funcionamento. Para cada uma, explique por que ela pode ou não ser a causa real.

(a) Existe uma conexão aberta da saída do inversor do LSB para as entradas D dos FFs.

(b) A entrada D do flip-flop Q_8 está internamente em curto-circuito com V_{CC}.

(c) Uma ponte de solda está colocando $\overline{Q}_0$ em curto-circuito com a terra.

T

9-21. Um estudante testa o circuito da Fig. 9-4, conforme descrito no Exemplo 9-7, e obtém os seguintes resultados: todas as saídas funcionam, exceto $\overline{O_{16}}$ a $\overline{O_{19}}$ e $\overline{O_{24}}$ a $\overline{O_{27}}$, que estão permanentemente em nível ALTO. Qual é a falha mais provável do circuito?

TABELA 9-7

Código de Entrada $A_4A_3A_2A_1A_0$	Saídas Ativadas
1 0 0 0 0	$\overline{O_{16}}$ e $\overline{O_{24}}$
1 0 0 0 1	$\overline{O_{17}}$ e $\overline{O_{25}}$
1 0 0 1 0	$\overline{O_{18}}$ e $\overline{O_{26}}$
1 0 0 1 1	$\overline{O_{19}}$ e $\overline{O_{27}}$
1 0 1 0 0	$\overline{O_{20}}$ e $\overline{O_{28}}$
1 0 1 0 1	$\overline{O_{21}}$ e $\overline{O_{29}}$
1 0 1 1 0	$\overline{O_{22}}$ e $\overline{O_{30}}$
1 0 1 1 1	$\overline{O_{23}}$ e $\overline{O_{31}}$

T

9-22. Um estudante testa o circuito da Fig. 9-4, conforme descrito no Exemplo 9-7, e verifica que a saída correta é ativada para cada um dos possíveis códigos de entrada, exceto para os que estão listados na Tabela 9-7. Examine esta tabela e determine o provável motivo do mau funcionamento.

T

9-23. Suponha que um resistor de 22 Ω seja usado erradamente para o segmento g na Fig. 9-8. Como isto afetaria o display? Que possíveis problemas poderiam ocorrer?

T

9-24. Repita o Exemplo 9-8 com a seqüência observada a seguir.

CONTAGEM	0	1	2	3	4	5	6	7	8	9
Display observado	0	1	2	3	8	9	c	ɔ	4	5

T

9-25. Repita o Exemplo 9-8 com a seqüência observada a seguir.

CONTAGEM	0	1	2	3	4	5	6	7	8	9
Display observado	0	7	2	3	9	9	8	7	8	9

T

9-26. Para testar o circuito da Fig. 9-11, uma estudante conectou um contador BCD nas entradas do 74HC4511 e injetou pulsos no contador a uma taxa muito lenta. Ela notou que o segmento f funcionou de modo errático, e nenhum padrão especial ficou evidente. Quais são algumas possíveis causas do mau funcionamento? (*Sugestão:* Lembre-se, os CIs são CMOS.)

SEÇÕES 9-7 E 9-8

9-27. O circuito na Fig. 9-61 utiliza três multiplexadores de duas entradas (Fig. 9-21). Determine a função realizada por este circuito.

D

9-28. Utilize a idéia do Problema 9-27 para interligar diversos multiplexadores 8 para 1, 74151, para formar um multiplexador 64 para 1.

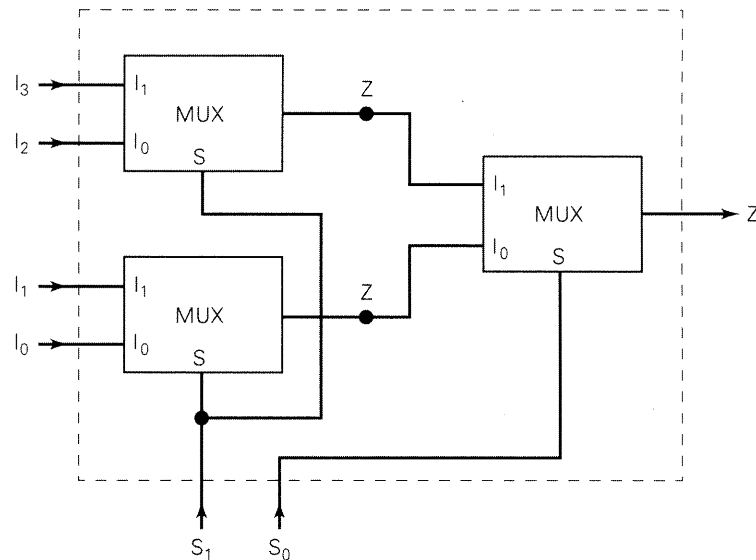


Fig. 9-61 Problema 9-27.

C, D

9-29. Mostre como dois 74157s e um 74151 podem ser conectados para formarem um multiplexador 16 para 1 sem nenhuma outra lógica. Identifique as entradas I_0 a I_{15} para mostrar como elas correspondem ao código de seleção.

D

9-30. (a) Expanda o circuito da Fig. 9-26 para mostrar o conteúdo de dois contadores BCD de três estágios.

(b) Conte o número de conexões neste circuito e compare com o número necessário, se um conjunto decodificador/driver com display separado fosse usado para cada estágio de cada contador.

9-31. A Fig. 9-62 mostra como um multiplexador pode ser utilizado para gerar formas de onda lógicas com qualquer padrão desejado. O padrão é programado usando oito chaves, e a forma de onda é produzida repetidamente acionando-se o contador de módulo 8. Desenhe a forma de onda de Z para a posição das chaves na figura.

9-32. Substitua o contador de módulo 8 na Fig. 9-62 por um contador de módulo 16 e conecte o MSB na entrada $\overline{E}$ do multiplexador. Desenhe a forma de onda de Z.

D

9-33. Mostre como um 74151 pode ser usado para gerar a função lógica $Z = AB + BC + AC$.

D

9-34. Mostre como um multiplexador de 16 entradas, tal como o 74150 (Fig. 9-37), deve ser usado para gerar a função $Z = \overline{A}\overline{B}\overline{C}D + BCD + A\overline{B}\overline{D} + AB\overline{C}D$.

N

9-35. O circuito da Fig. 9-63 mostra como um MUX de oito entradas pode ser utilizado para gerar uma função lógica de quatro variáveis, embora o MUX tenha apenas três entradas de seleção. Três das variáveis lógicas, A, B e C, estão conectadas nas entradas de seleção. A quarta variável D e seu complemento $\overline{D}$ estão conectados em determinadas entradas de dados do MUX, conforme é necessário para a função lógica desejada. As outras entradas de dados do MUX estão em BAIXO ou em ALTO conforme a função necessita.

(a) Construa uma tabela-verdade mostrando a saída Z para as 16 combinações possíveis das variáveis de entrada.

(b) Escreva a expressão da soma de produtos para Z e simplifique-a para verificar que

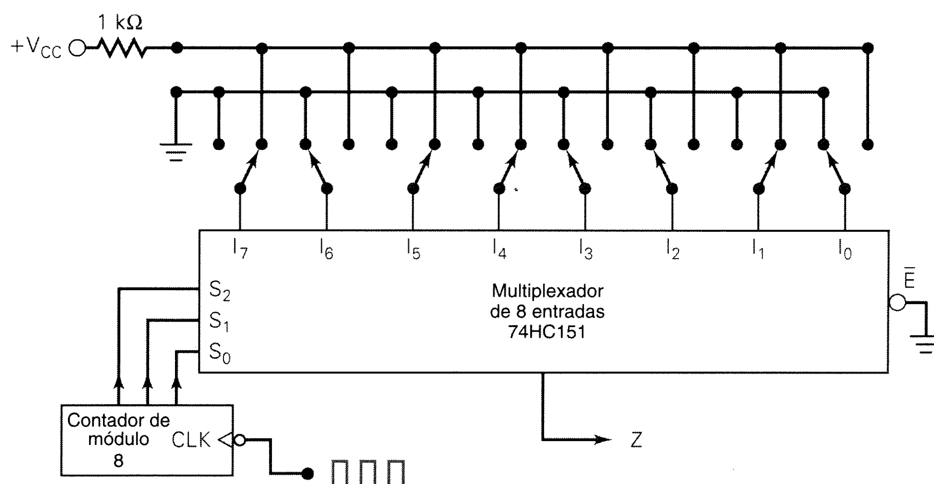


Fig. 9-62 Problemas 9-31 e 9-32.

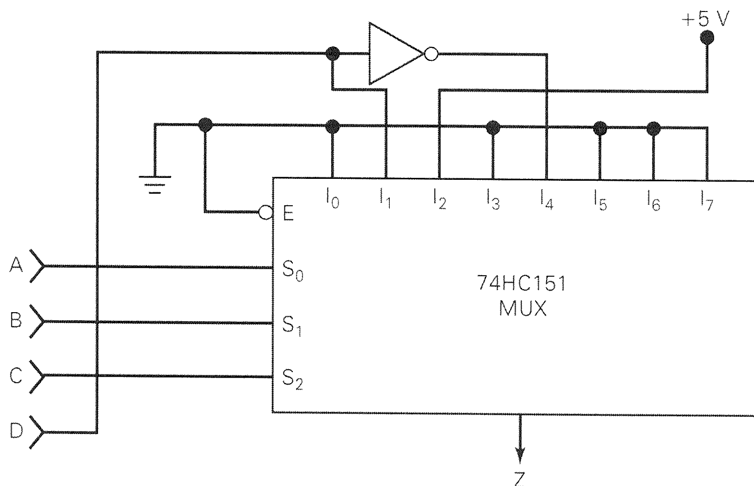


Fig. 9-63 Problemas 9-35 e 9-36.

$$Z=\overline{C}B\overline{A}+D\overline{C}\,\overline{B}A+\overline{D}C\overline{B}\,\overline{A}$$

C, D

9-36. O método usado na Fig. 9-63 pode ser utilizado para gerar qualquer função lógica de quatro variáveis seguindo-se estes passos:

1. Construa a tabela-verdade para a função desejada com saída Z.

2. Escreva a expressão da soma de produtos para Z e não a simplifique. Por exemplo,

$$Z=DC\overline{B}A+\overline{D}CB\overline{A}+DCB\overline{A}+D\overline{C}BA+\overline{D}\,\overline{C}BA+\overline{D}\,\overline{C}\,\overline{B}A.$$

3. Procure por termos que tenham a mesma combinação de C, B e A, e fatore:

$$\begin{aligned} Z &= DC\overline{B}A + CB\overline{A}\,(\overline{D}+D) + \overline{C}BA(\overline{D}+D) + \overline{D}\,\overline{C}\,\overline{B}A \\ &= DC\overline{B}A + CB\overline{A} + \overline{C}BA + \overline{D}\,\overline{C}\,\overline{B}A \end{aligned}$$

4. Considere os termos que contiverem somente C, B e A, na forma normal ou na complementar. Para cada um deles, conecte a entrada de dados correspondente do MUX em ALTO:

$CB\overline{A} \rightarrow$ conecte I_6 em ALTO

$\overline{C}BA \rightarrow$ conecte I_3 em ALTO

5. Considere os termos que têm a variável D. Conecte a variável D ou $\overline{D}$ na entrada do MUX que corresponde às variáveis CBA.

$DC\overline{B}A \rightarrow$ conecte D na entrada I_5

$\overline{D}\,\overline{C}\,\overline{B}A \rightarrow$ conecte D na entrada I_1

6. Conecte as entradas restantes do MUX em BAIXO.

(a) Verifique o projeto da Fig. 9-63 utilizando este método.

(b) Use este método para implementar uma função que produz um nível ALTO somente quando as quatro variáveis de entrada estiverem no mesmo nível ou quando as variáveis B e C estiverem em níveis diferentes.

SEÇÃO 9-9

9-37. QUESTÃO DE FIXAÇÃO

Para cada sentença, indique se ela está se referindo a um decodificador, a um codificador, a um MUX ou a um DEMUX.

(a) Tem mais entradas do que saídas.

(b) Utiliza entradas de SELEÇÃO.

(c) Pode ser usado em conversão paralelo-série.

(d) Produz um código binário na sua saída.

(e) Somente uma de suas saídas pode estar ativa por vez.

(f) Pode ser usado para rotear um sinal de entrada para uma das várias saídas possíveis.

(g) Pode ser utilizado para gerar funções lógicas arbitrárias.

9-38. Mostre como o decodificador 7442 pode ser usado como um demultiplexador 8 para 1. (*Sugestão:* Veja o Problema 9-7.)

9-39. Aplique as formas de onda da Fig. 9-59 nas entradas do DEMUX 74LS138 da Fig. 9-32(a) da seguinte maneira:

$D \rightarrow A_2 \qquad C \rightarrow A_1 \qquad B \rightarrow A_0 \qquad A \rightarrow \overline{E}_1$

Desenhe as formas de onda das saídas do DEMUX.

9-40. Considere o sistema da Fig. 9-34. Admita que a freqüência do clock é 10 pps. Descreva quais serão as indicações do painel de monitoração para cada um dos seguintes casos:

(a) Todas as portas fechadas.

(b) Todas as portas abertas.

(c) Portas 2 e 6 abertas.

C, D

9-41. Modifique o sistema da Fig. 9-34 para gerenciar 16 portas. Utilize um MUX de 16 entradas 74150 e dois DEMUXes 74LS138. Quantas linhas irão para o painel remoto de monitoração?

9-42. Desenhe as formas de onda em Z, O_0, O_1, O_2 e O_3 na Fig. 9-35 para os seguintes dados nos registradores: $[A]$ = 0011, $[B]$ = 0110, $[C]$ = 1001 e $[D]$ = 0111.

SEÇÃO 9-11

T

9-43. Considere o seqüenciador de controle da Fig. 9-28. Descreva como cada uma das seguintes falhas vai afetar a sua operação.

(a) A entrada I_3 do MUX está em curto com a terra.

(b) As conexões dos sensores 3 e 4 para o MUX estão trocadas.

T

9-44. Considere o circuito da Fig. 9-26. Um teste do circuito levou aos resultados mostrados na Tabela 9-8. Quais são as possíveis causas de mau funcionamento?

T

9-45. Um teste do circuito de monitoração de segurança da Fig. 9-34 produziu os resultados registrados na Tabela 9-9. Quais

TABELA 9-8

		Contagem Real	Contagem Apresentada
Caso 1	Contador 1	33	33
	Contador 2	47	47
Caso 2	Contador 1	82	02
	Contador 2	64	64
Caso 3	Contador 1	63	63
	Contador 2	95	15

TABELA 9-9

Condição	LEDs
Todas as portas fechadas	Todos os LEDs apagados
Porta 0 aberta	LED 0 piscando
Porta 1 aberta	LED 2 piscando
Porta 2 aberta	LED 1 piscando
Porta 3 aberta	LED 3 piscando
Porta 4 aberta	LED 4 piscando
Porta 5 aberta	LED 6 piscando
Porta 6 aberta	LED 5 piscando
Porta 7 aberta	LED 7 piscando

são as possíveis falhas que poderiam provocar esta operação?

T

9-46. Um teste do circuito de monitoração de segurança da Fig. 9-34 produziu os resultados registrados na Tabela 9-10. Quais são as possíveis falhas que poderiam provocar esta operação? Como isto pode ser verificado ou eliminado como uma falha?

TABELA 9-10

Condição	LEDs
Todas as portas fechadas	Todos os LEDs apagados
Porta 0 aberta	LED 0 piscando
Porta 1 aberta	LED 1 piscando
Porta 2 aberta	LED 2 piscando
Porta 3 aberta	LED 3 piscando
Porta 4 aberta	LED 4 piscando
Porta 5 aberta	LED 5 piscando
Porta 6 aberta	Nenhum LED piscando
Porta 7 aberta	Nenhum LED piscando
Portas 6 e 7 abertas	LEDs 6 e 7 piscando

T

9-47. O sistema síncrono de transmissão de dados da Fig. 9-35 não está funcionando bem. Um osciloscópio é utilizado para monitorar as saídas do MUX e do DEMUX durante o ciclo de transmissão com os resultados mostrados na Fig. 9-64. Quais são as possíveis causas do mau funcionamento?

T

9-48. O sistema síncrono de transmissão de dados da Fig. 9-35 não está funcionando bem e as formas de onda foram pesquisadas com um osciloscópio de alta performance (Fig. 9-65). Note os glitches no sinal O_1. Considere as duas falhas possíveis relacionadas a seguir. Para cada uma delas, explique se ela pode ou não ser a causa do mau funcionamento.

(a) As conexões dos pinos S_1 e S_0 do DEMUX estão trocadas.

(b) As conexões para os pinos Q_1 e Q_0 do contador de palavras do receptor estão trocadas.

SEÇÃO 9-12

C, D

9-49. Projete novamente o circuito do Problema 9-16 utilizando um comparador de magnitude 74HC85. Inclua uma característica "detecção de excesso de cópias", que ativará uma saída ALARME se a saída OPERAÇÃO falhar em parar a máquina quando o número requisitado de cópias for atingido.

D

9-50. Mostre como conectar 74HC85s para comparar dois números de 10 bits.

SEÇÃO 9-13

9-51. Admita uma entrada BCD 69 para o conversor de códigos da Fig. 9-45. Determine os níveis de cada saída Σ e da saída binária final.

T

9-52. Um estudante testa o conversor de códigos da Fig. 9-45 e observa os seguintes resultados:

Entrada BCD	*Saída Binária*
52	0110011
95	1100000
27	0011011

Qual é a falha provável do circuito?

SEÇÕES 9-14 A 9-16

9-53. QUESTÃO DE FIXAÇÃO

Verdadeiro ou falso:

(a) Um dispositivo conectado num barramento de dados deveria ter saídas tristate.

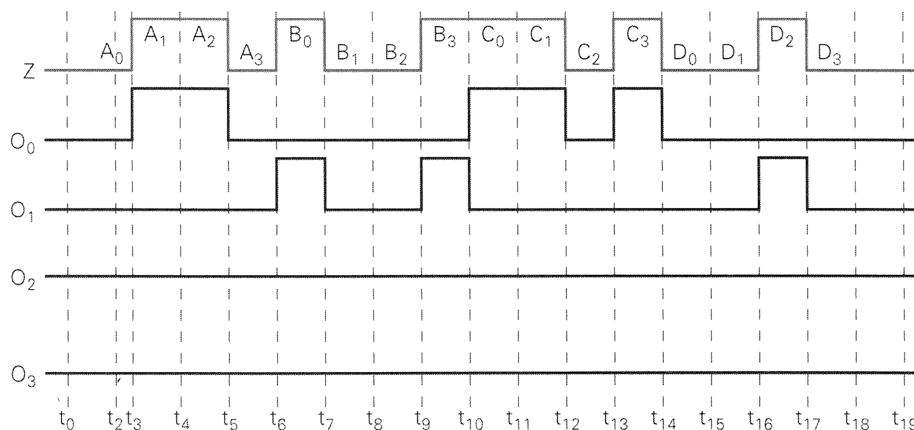


Fig. 9-64 Problema 9-47.

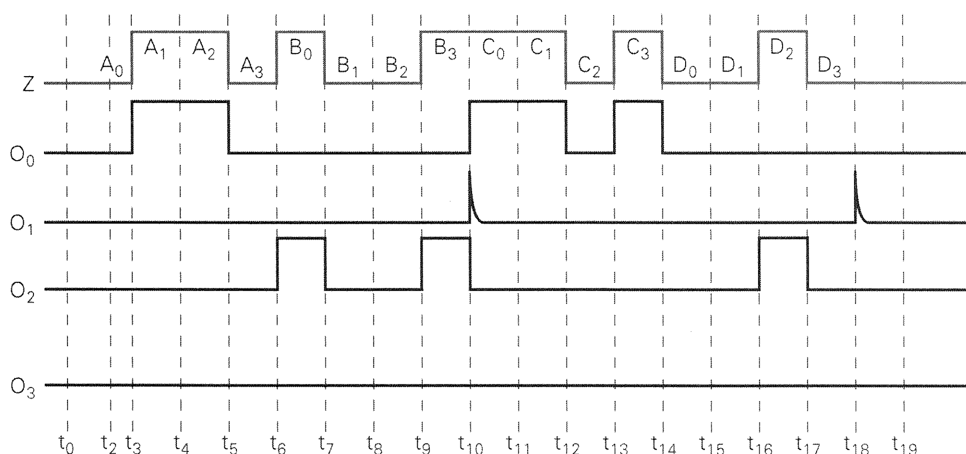


Fig. 9-65 Problema 9-48.

(b) Contenção de barramento ocorre quando mais do que um dispositivo pega um dado do barramento.

(c) Dados de maior largura podem ser transferidos por um barramento de oito linhas do que por um barramento de quatro linhas.

(d) Um CI driver de barramento geralmente tem uma alta impedância de saída.

(e) Registradores e buffers bidirecionais têm linhas de I/O comuns.

9-54. Para a configuração de barramento da Fig. 9-49, descreva os sinais de entrada necessários para transferir simultaneamente o conteúdo do registrador *C* para os outros registradores.

9-55. Admita que os registradores na Fig. 9-49 estão inicialmente com [*A*] = 1011, [*B*] = 1000 e [*C*] = 0111. Os sinais na Fig. 9-66 são aplicados nos registradores.

(a) Determine os conteúdos de cada registrador nos instantes t_1, t_2, t_3 e t_4.

(b) Descreva o que aconteceria se $\overline{IE_A}$ estivesse em BAIXO quando o terceiro pulso de clock ocorresse.

9-56. Admita as mesmas condições iniciais do Problema 9-55 e desenhe o sinal em DB_3 para as formas de onda da Fig. 9-66.

9-57. A Fig. 9-67 mostra dois dispositivos extras que devem ser incluídos no barramento de dados da Fig. 9-49. Um deles é um conjunto de chaves com buffers, que podem ser usadas para entrada manual de dados em qualquer dos registradores do barramento. O outro dispositivo é um registrador de saída, que é utilizado para armazenar qualquer dado que esteja no barramento durante uma operação de transferência de dados e apresentá-lo num conjunto de LEDs.

(a) Suponha que todos os registradores contêm 0000. Faça um resumo da seqüência de operações necessárias para carregar os registradores com os seguintes dados pelas chaves:
[*A*] = 1011, [*B*] = 0001, [*C*] = 1110.

(b) Qual será o estado dos LEDs no final desta seqüência?

C

9-58. Agora que o circuito da Fig. 9-67 foi incluído na Fig. 9-49, um total de cinco dispositivos está conectado no barramento de dados. O circuito na Fig. 9-68(a) será usado então para gerar os sinais de habilitação necessários para realizar as diferentes transferências de dados pelo barramento. Ele utiliza um chip 74HC139 que contém dois decodificadores 2 para 4 idênticos e independentes com uma habilitação ativa em BAIXO. O decodificador superior é usado para selecio-

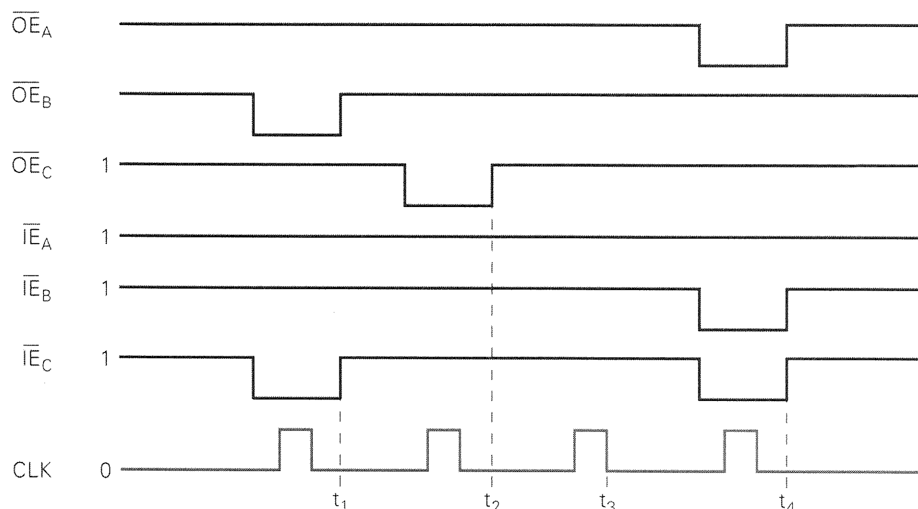


Fig. 9-66 Problemas 9-55 e 9-56.

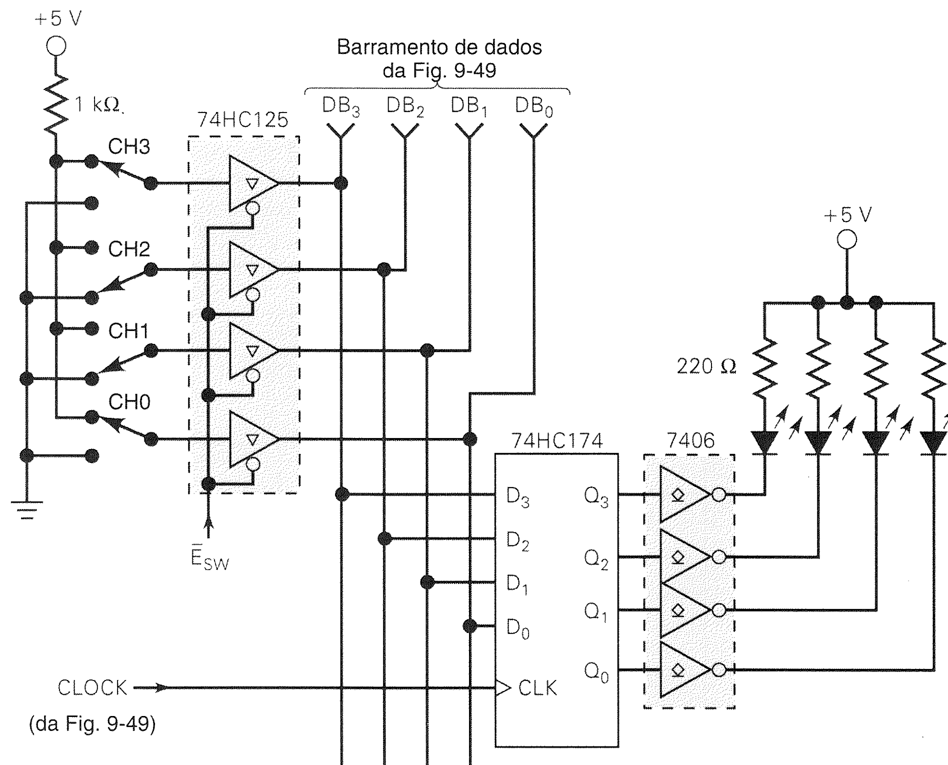


Fig. 9-67 Problemas 9-57, 9-58 e 9-59.

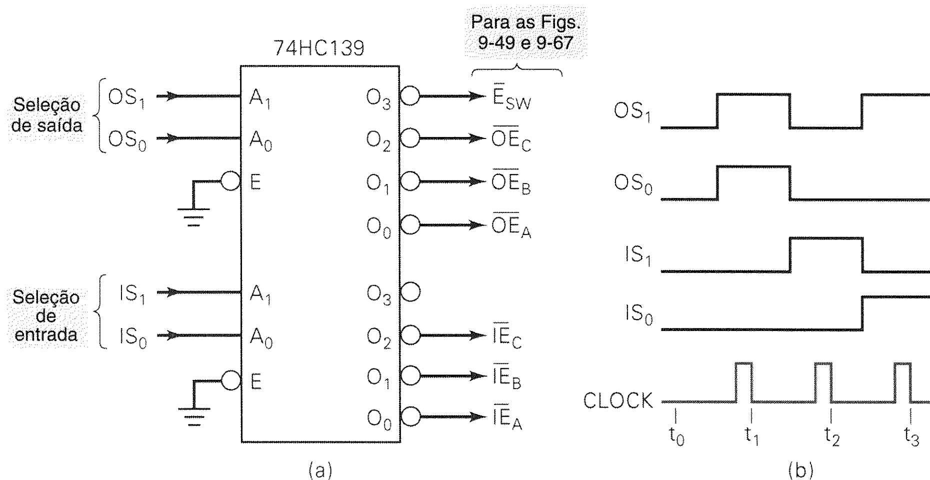


Fig. 9-68 Problema 9-58.

nar o dispositivo que vai colocar o dado no barramento (seleção de saída), e o decodificador inferior é utilizado para selecionar o dispositivo que vai pegar o dado do barramento (seleção de entrada). Suponha que as saídas do decodificador sejam conectadas nas entradas de habilitação correspondentes dos dispositivos ligados ao barramento. Suponha também que todos os registradores estão inicialmente com 0000 no instante t_0, e as chaves estão nas posições mostradas na Fig. 9-67.

(a) Determine o conteúdo de cada um dos registradores nos instantes t_1, t_2 e t_3 em resposta às formas de onda na Fig. 9-68(b).

(b) Pode ocorrer contenção de barramento neste circuito? Explique.

9-59. Mostre como um 74HC541 (Fig. 9-52) pode ser utilizado no circuito da Fig. 9-67.

APLICAÇÃO EM MICROCOMPUTADOR

C, N

9-60. A Fig. 9-69 mostra o circuito básico para interfacear um microprocessador com um módulo de memória. O módulo de memória contém um ou mais CIs de memória (Cap. 11) que podem tanto receber dados do microprocessador (operação de ESCRITA) como enviar dados para o microproces-

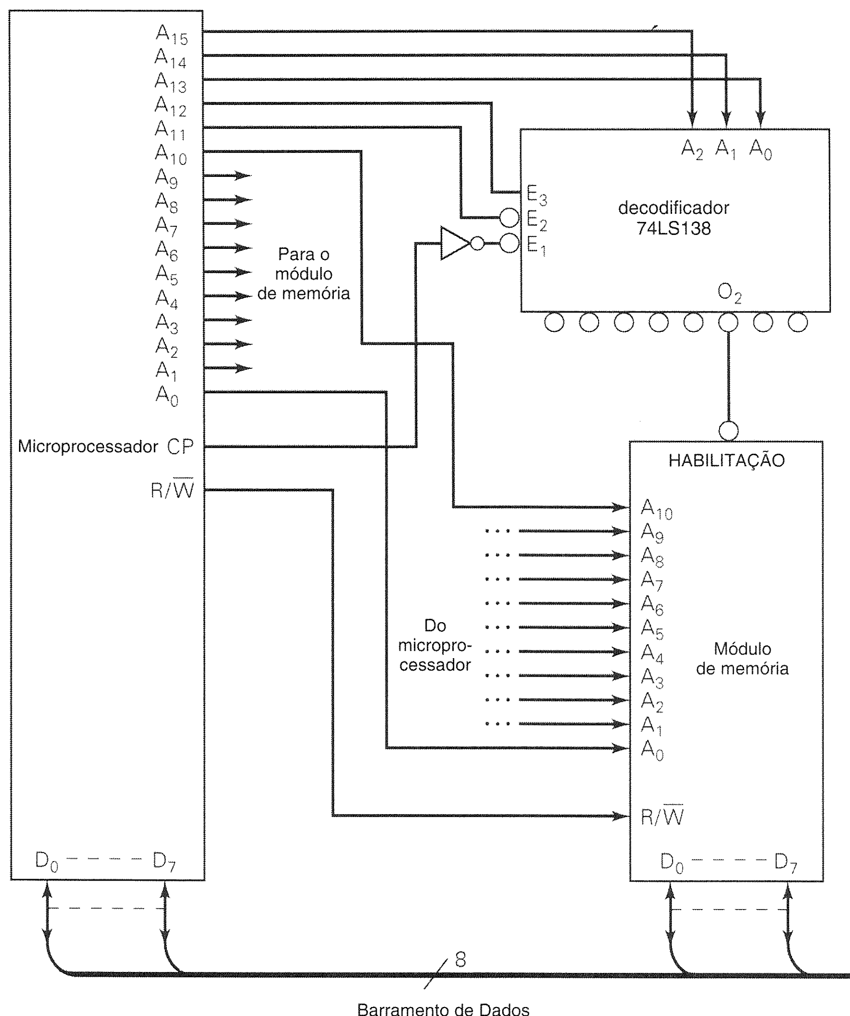


Fig. 9-69 Circuito básico de interface entre microprocessador e memória para o Problema 9-60.

sador (operação de LEITURA). O dado é transferido pelas oito linhas do barramento de dados. As linhas de dados do microprocessador e as linhas de dados da memória são conectadas nesta barra comum. No momento, vamos analisar como o microprocessador controla a seleção do módulo de memória para uma operação de LEITURA ou de ESCRITA.

Os seguintes passos estão envolvidos:

1. O microprocessador coloca o endereço de memória nas suas linhas de saída de endereço A_{15} a A_0.
2. O microprocessador gera o sinal $R/\overline{W}$ para informar ao módulo de memória que operação deve ser realizada: $R/\overline{W} = 1$ para LEITURA e $R/\overline{W} = 0$ para ESCRITA.
3. Os cinco bits mais significativos das linhas de endereço do microprocessador são decodificadas pelo 74LS138, que controla a entrada de ENABLE (HABILITAÇÃO) do módulo de memória. Esta entrada de ENABLE deve estar ativa para que o módulo de memória possa realizar operações de leitura ou de escrita.
4. Os outros 11 bits de endereço são conectados no módulo de memória, que os utiliza para selecionar a posição *interna* de memória que está sendo acessada pelo microprocessador, desde que ENABLE esteja ativo.

Para ler ou escrever no módulo de memória, o microprocessador deve colocar o endereço correto nas linhas de endereço para habilitar a memória, e então pulsar *CP* no estado ALTO.

(a) Determine quais, caso exista algum, destes endereços de memória ativarão o módulo de memória: 607F, 57FA e 5F00.

(b) Determine a faixa de endereços em hexadecimal que ativará a memória. (*Sugestão:* As entradas A_{10} até A_0 para a memória podem ter qualquer combinação.)

(c) Admita que um segundo módulo de memória idêntico é adicionado ao circuito com seus sinais de endereços, $R/\overline{W}$, e linhas de dados conectados exatamente do mesmo modo que o primeiro módulo, *exceto* que sua entrada de ENABLE está ligada na saída $\overline{O_4}$. Que faixa de endereços em hexadecimal vai ativar este segundo módulo?

(d) É possível o microprocessador ler ou escrever de ambos os módulos ao mesmo tempo? Explique.

PROBLEMA DE PROJETO

C, D

9-61. O circuito de teclado da Fig. 9-16 será usado como parte de uma tranca eletrônica digital que opera do seguinte modo: quando ativada, uma saída DESTRAVA vai para ALTO. Este nível ALTO é utilizado para energizar um solenóide que retrai um pino e permite que a porta seja aberta. Para ativar DESTRAVA, o operador deve pressionar a tecla LIMPA e então digitar a seqüência correta de três dígitos.

(a) Mostre de que maneira comparadores 74HC85 com mais alguma lógica necessária podem ser adicionados ao circuito de teclado para produzir a operação da tranca digital descrita anteriormente para uma seqüência de entrada de LIMPA-3-5-8.
(b) Modifique o circuito para ativar uma saída de ALARME se o operador digitar algo diferente do que a seqüência correta de três dígitos.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 9-1
1. Não
2. A entrada de habilitação controla se a lógica do decodificador responde ou não ao código binário de entrada.
3. O 7445 tem saídas em coletor aberto que podem suportar até 30 V e 80 mA.
4. 24 pinos: 2 de habilitação, 4 entradas, 16 saídas, V_{CC} e terra.

SEÇÃO 9-2
1. *a, b, c, f, g*
2. Verdadeiro

SEÇÃO 9-3
1. LEDs: (a), (e), (f). LCDs: (b), (c), (d), (e)
2. (a) BCD de 4 bits
(b) ASCII de 7 ou 8 bits
(c) valor binário para a intensidade do pixel

SEÇÃO 9-4
1. Um codificador produz um código de saída correspondente à entrada ativada. Um decodificador ativa uma saída correspondente a um código aplicado na entrada.
2. Em um codificador de prioridade, o código de saída corresponde à entrada de número *mais elevado* que for ativada.
3. BCD normal = 0110
4. (a) Produz uma transição positiva quando uma tecla é pressionada.
(b) Converte uma atuação de tecla em seu código BCD.
(c) Gera pulsos sem os efeitos da trepidação de contato para acionar o contador.
(d) Formam um contador em anel que aciona seqüencialmente os registradores de saída.
(e) Armazenam os códigos BCD gerados pelas atuações das teclas.

SEÇÃO 9-5
1. Decodificador BCD para decimal
2. Alta prioridade
3. Buffer ou driver

SEÇÃO 9-7
1. O número binário nas entradas de seleção determina que entrada de dados vai passar para a saída.
2. Trinta e duas entradas de dados e cinco entradas de seleção.

SEÇÃO 9-8
1. Conversão paralelo-série, roteamento de dados, geração de funções lógicas e seqüenciamento de operações.
2. Falso; elas são aplicadas nas entradas de seleção.
3. Contador

SEÇÃO 9-9
1. Um MUX seleciona um dentre vários sinais de entrada para ser passado para sua saída; um DEMUX seleciona uma das várias saídas para receber o sinal de entrada.
2. Verdadeiro, desde que o decodificador tenha uma entrada de habilitação.
3. Os LEDs vão acender e apagar em seqüência.

SEÇÃO 9-12
1. Para fornecer um meio de expandir a operação de comparação para números com mais do que quatro bits.
2. $O_{A=B}$ = 1; as outras saídas são 0.

SEÇÃO 9-13
1. Um conversor de código recebe sua entrada de dados representada em um tipo de código binário e a converte para outro tipo de código binário.
2. Três dígitos podem representar valores decimais até 999. Para representar 999 em binário puro, necessita-se de 10 bits.

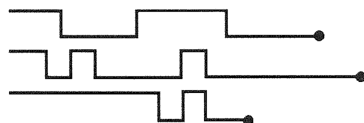
SEÇÃO 9-14
1. Um conjunto de linhas nas quais as entradas e saídas de muitos dispositivos diferentes podem ser conectadas.
2. Contenção de barramento ocorre quando saídas de mais do que um dispositivo conectado ao barramento são habilitadas ao mesmo tempo. É evitada controlando-se as entradas de habilitação dos dispositivos para que isto não aconteça.
3. Uma situação na qual todos os dispositivos conectados ao barramento estão em alta impedância.

SEÇÃO 9-15
1. 1011
2. Verdadeiro
3. 0000.

SEÇÃO 9-16
1. Contenção de barramento
2. Em flutuação, alta impedância
3. Fornecer saídas tristate de baixa impedância
4. Reduzir o número de pinos do CI e o número de conexões para o barramento
5. Veja a Fig. 9-56.

CAPÍTULO 10

Interface com o Mundo Analógico



■ SUMÁRIO

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Compreender a teoria de operação e as limitações dos circuitos de diversos tipos de conversores digital-analógicos (D/A).
- Ler e entender as várias especificações dos fabricantes de conversores D/A.
- Utilizar diferentes procedimentos de teste para pesquisar falhas em circuitos de conversores D/A.
- Comparar as vantagens e desvantagens entre os conversores analógico-digitais (A/D) de rampa digital, por aproximações sucessivas e do tipo flash.
- Analisar o processo pelo qual um computador em conjunto com um conversor A/D digitaliza um sinal analógico e então o reconstitui a partir dos dados digitais.
- Descrever a operação básica de um voltímetro digital.
- Compreender a necessidade de utilização de circuitos de amostragem e retenção (sample-and-hold) em conjunto com os conversores A/D.
- Descrever a operação de um sistema de multiplexação analógico.
- Entender as características e a operação básica de um osciloscópio de memória digital.
- Compreender os conceitos básicos de processamento digital de sinais.

10-1 INTERFACE COM O MUNDO ANALÓGICO

Revisão de Digital versus Analógico

Uma **quantidade digital** terá um valor que é especificado entre duas possibilidades, tal como 0 ou 1, BAIXO ou ALTO, falso ou verdadeiro e assim por diante. Na prática, uma quantidade digital como uma tensão pode ter, na verdade, um valor que está dentro de faixas especificadas, e definimos que valores dentro de uma determinada faixa possuem o mesmo valor digital. Por exemplo, para a lógica TTL sabemos que

0 V a 0,8 V = 0 lógico
2 V a 5 V = 1 lógico

Para qualquer tensão na faixa de 0 a 0,8 V atribui-se o valor digital 0, e para qualquer tensão na faixa de 2 a 5 V associa-se o valor digital 1. Os valores exatos de tensão não são importantes, porque os circuitos digitais respondem da mesma maneira para todas as tensões dentro de determinada faixa.

Por outro lado, uma **quantidade analógica** pode assumir qualquer valór de uma faixa contínua de valores, e, mais importante, seu valor exato é significativo. Por exemplo, a saída de um conversor analógico temperatura-tensão é medida como 2,76 V, o que pode representar uma determinada temperatura de 27,6°C. Se a tensão medida for diferente, tal como 2,34 V ou 3,78 V, isto representaria uma temperatura completamente diferente. Em outras palavras, cada valor possível de uma quantidade analógica tem um significado diferente. Um outro exemplo é a tensão de saída de um amplificador de áudio para um alto-falante. Esta tensão é uma quantidade analógica porque cada um dos seus valores possíveis produz uma resposta diferente no alto-falante.

A maioria das variáveis físicas é analógica e pode assumir qualquer valor dentro de uma faixa contínua de valores. Podemos citar como exemplos: temperatura, pressão, intensidade luminosa, sinais de áudio, posição, velocidade de rotação e velocidade de fluxo. Sistemas digitais realizam todas as suas operações internas utilizando circuitos e operações digitais. Qualquer informação que deva entrar em um sistema digital precisa ser colocada primeiro na forma digital. Analogamente, as saídas de um sistema digital sempre estão na forma digital. Quando um sistema digital, como um computador, é usado para monitorar e/ou controlar um processo físico, devemos lidar com as diferenças entre a natureza digital dos computadores e a natureza analógica das variáveis do processo. A Fig. 10-1 ilustra esta situação. Este diagrama mostra os cinco elementos que estão envolvidos quando um computador está monitorando e controlando uma variável física, que admitimos ser analógica:

1. **Transdutor.** Normalmente, a variável física não é uma quantidade elétrica. Um **transdutor** é um dispositivo que converte uma variável física em uma variável elétrica. Alguns transdutores comuns são: termistores, fotocélulas, fotodiodos, medidores de fluxo, transdutores de pressão e tacômetros. A saída elétrica de um transdutor é uma corrente ou uma tensão analógica que é proporcional à variável física que está monitorando. Por exemplo, a variável física poderia ser a temperatura da água num grande tanque que está sendo abastecido por canos de água quente e água fria. Vamos imaginar que a temperatura da água varia de 25°C até 65°C e que um termistor e seus circuitos associados convertem estes valores de temperatura da água para uma tensão na faixa de 250 a 650 mV. Repare que a saída do transdutor é diretamente proporcional à temperatura, de tal modo que cada 1°C produz uma saída de 10 mV. Este fator de proporcionalidade foi escolhido por conveniência.
2. **Conversor analógico-digital (A/D).** A saída analógica do transdutor serve de entrada analógica para o conversor A/D. O conversor A/D converte esta entrada analógica em uma saída digital. Esta saída digital são os bits que representam o valor da entrada analógica. Por exemplo, o conversor A/D poderia converter os valores analógicos do transdutor, 250 a 650 mV, para valores binários na faixa de 00011001(25) a 01000001(65). Note que a saída binária do conversor A/D é proporcional à tensão de entrada analógica, de modo que cada unidade da saída digital representa 10 mV.
3. **Computador.** A representação digital da variável do processo é transmitida do A/D para o computador digital, que armazena este valor digital e o processa de acordo com as instruções de um programa que está execu-

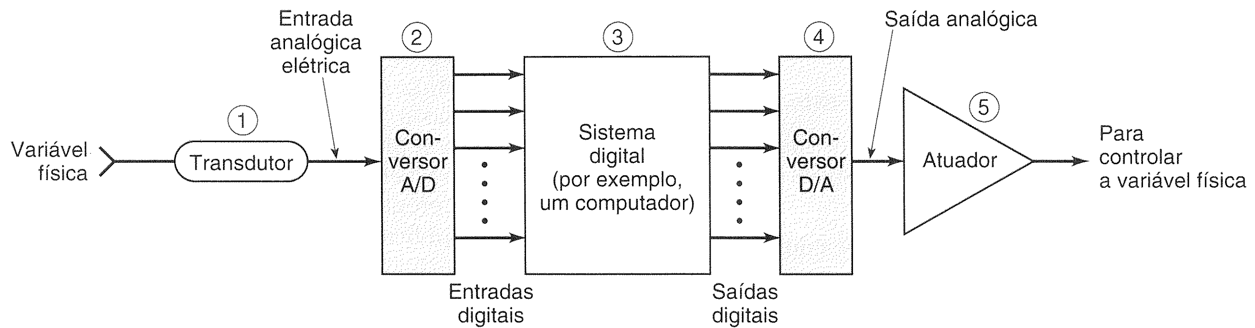


Fig. 10-1 Conversores do tipo analógico-digital (A/D) e digital-analógico (D/A) são utilizados para interfacear um computador com o mundo analógico, de modo que o computador possa monitorar e controlar uma variável física.

tando. O programa pode realizar cálculos ou outras operações com esta representação digital da temperatura para gerar uma saída digital que eventualmente será utilizada para controlar a temperatura.

4. **Conversor digital-analógico (D/A).** Esta saída digital do computador é conectada em um conversor D/A, que, por sua vez, a converte para uma tensão ou corrente analógica proporcional. Por exemplo, o computador pode produzir uma saída digital variando desde 00000000 até 11111111, que o conversor D/A converte para uma tensão na faixa de 0 a 10 V.
5. **Atuador.** O sinal analógico do conversor D/A freqüentemente é conectado a algum dispositivo ou circuito que serve como um atuador para controlar a variável física. Para o nosso exemplo de temperatura da água, o atuador poderia ser uma válvula controlada eletricamente que regulasse o fluxo de água quente para o tanque, de acordo com a tensão analógica do conversor D/A. O fluxo variaria de modo proporcional a esta tensão analógica, com 0 V produzindo nenhum fluxo e 10 V produzindo o fluxo máximo.

Pelo que foi apresentado, conclui-se que os conversores A/D e D/A funcionam como *interfaces* entre um sistema completamente digital, como um computador, e o mundo analógico. Esta função se torna cada vez mais importante à medida que microcomputadores de baixo custo têm sido utilizados em áreas de controle de processos em que antes o controle computacional não era justificável.

Questões de Revisão

1. Qual é a função de um transdutor?
2. Qual é a função de um conversor A/D?
3. O que um computador normalmente faz com o dado que recebe do conversor A/D?
4. Que função é realizada por um conversor D/A?
5. Qual é a função de um atuador?

10-2 CONVERSÃO DIGITAL-ANALÓGICA (D/A)

Iniciaremos agora nosso estudo sobre as conversões digital-analógica (D/A) e analógica-digital (A/D). Tendo em vista que muitos métodos de conversão A/D utilizam o processo de conversão D/A, examinaremos primeiramente a conversão D/A.

Basicamente, a *conversão* D/A é o processo em que um valor representado em código *digital* (tal como o binário puro ou o BCD) é convertido para uma tensão ou corrente, que é proporcional ao valor digital. A Fig. 10-2(a) mostra o símbolo para um conversor D/A típico de quatro bits. Por enquanto, não vamos nos preocupar com os circuitos internos. No momento, examinaremos as diversas relações de entrada e saída.

Note que existe uma entrada para uma tensão de referência, V_{ref}. Esta entrada é utilizada para determinar a **saída de fundo de escala** ou o máximo valor que o conversor D/A pode produzir. As entradas digitais *D*, *C*, *B* e *A* normalmente são derivadas de um registrador de saída de um sistema digital. Os $2^4 = 16$ números binários diferentes representados por estes quatro bits estão relacionados na Fig. 10-2(b). Para cada número de entrada, a tensão de saída do conversor D/A é um valor único. Na verdade, para este caso, a tensão analógica de saída V_{OUT} é igual em volts ao número binário. Também poderia ser duas vezes o número binário ou ter outro fator de proporcionalidade. A mesma idéia seria verdadeira se a saída do D/A fosse uma corrente I_{OUT}.

Em geral,

$$\textbf{saída analógica} = K \times \textbf{entrada digital} \qquad (10\text{-}1)$$

onde *K* é o fator de proporcionalidade e é um valor constante para um conversor D/A em particular conectado a uma tensão de referência fixa. Naturalmente, a saída analógica pode ser uma tensão ou uma corrente. Quando for uma tensão, *K* estará em unidades de tensão, e, quando for uma corrente, *K* estará em unidades de corrente. Para o conversor D/A da Fig. 10-2, $K = 1$ V, de modo que

$$V_{OUT} = (1\text{ V}) \times \text{entrada digital}$$

Podemos utilizar isto para calcular V_{OUT} para diversos valores da entrada digital. Por exemplo, com uma entrada digital de $1100_2 = 12_{10}$, obtemos

$$V_{OUT} = 1\text{ V} \times 12 = 12\text{ V}$$

EXEMPLO 10-1A

Um conversor D/A de cinco bits tem saída em corrente. Para uma entrada digital de 10100, uma corrente de saída de 10

mA é produzida. Qual será a corrente de saída, I_{OUT}, para uma entrada digital de 11101?

Solução

A entrada digital 10100_2 é igual a 20 em decimal. Já que I_{OUT} = 10 mA para este caso, o fator de proporcionalidade deve ser 0,5 mA. Logo, podemos determinar I_{OUT} para qualquer entrada digital, tal como $11101_2 = 29_{10}$, da seguinte forma:

$$I_{OUT} = (0{,}5 \text{ mA}) \times 29$$
$$= 14{,}5 \text{ mA}$$

Lembre-se de que o fator de proporcionalidade *K* varia de um conversor D/A para outro e depende da tensão de referência.

EXEMPLO 10-1B

Qual é o maior valor de tensão de saída de um conversor D/A de oito bits que produz 1,0 V para uma entrada digital de 00110010?

Solução

$$00110010_2 = 50_{10}$$
$$1{,}0 \text{ V} = K \times 50$$

Portanto,

$$K = 20 \text{ mV}$$

A maior saída vai ocorrer para uma entrada de $11111111_2 = 255_{10}$.

$$V_{OUT}(\text{max}) = 20 \text{ mV} \times 255$$
$$= 5{,}10 \text{ V}$$

Saída Analógica

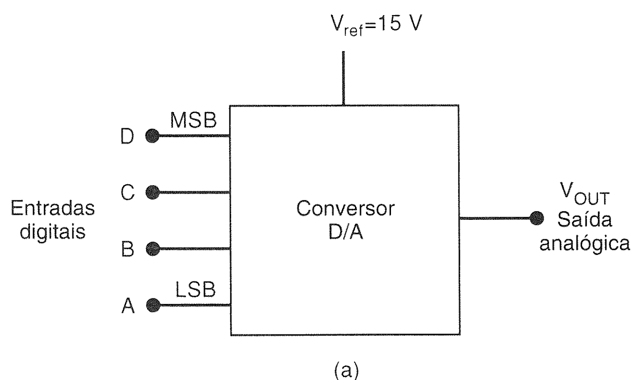
Tecnicamente, a saída de um conversor D/A não é uma quantidade analógica porque pode assumir apenas valores específicos, tais como os 16 níveis de tensão possíveis para V_{OUT} na Fig. 10-2 desde que V_{ref} seja constante. Assim, neste sentido, ela na verdade é digital. Entretanto, como veremos, o número de valores possíveis diferentes na saída pode ser aumentado e a diferença entre valores sucessivos diminuída, aumentando-se o número de bits de entrada. Isto nos permitirá produzir uma saída que é bastante parecida com uma quantidade analógica que varia continuamente dentro de uma faixa de valores. Em outras palavras, a saída do conversor D/A é uma quantidade "pseudo-analógica". Continuaremos a nos referir a ela como analógica, tendo em mente que é uma aproximação de uma quantidade analógica pura.

Pesos de Entrada

Para o conversor D/A da Fig. 10-2, deve-se notar que cada entrada digital contribui de forma diferente para a saída analógica. Isto é facilmente constatado se examinarmos os casos em que apenas uma entrada está em ALTO (Tabela 10-1). As contribuições de cada entrada digital são *ponderadas* de acordo com sua posição no número binário. Assim, *A*, que é o LSB, tem um *peso* de 1 V, *B* tem um peso de 2 V, *C* tem um peso de 4 V e *D*, o MSB, tem o maior peso, 8 V. Os pesos são sucessivamente dobrados para cada bit, começando com o LSB. Logo, podemos considerar V_{OUT} como sendo a soma ponderada das entradas digitais. Por exemplo, para determinar V_{OUT} para a entrada digital 0111, podemos somar os pesos dos bits *C*, *B* e *A* para obter 4 V + 2 V + 1 V = 7 V.

TABELA 10-1

D	*C*	*B*	*A*		V_{OUT} (V)
0	0	0	1	→	1
0	0	1	0	→	2
0	1	0	0	→	4
1	0	0	0	→	8



D	C	B	A	V_{OUT}	
0	0	0	0	0	volts
0	0	0	1	1	
0	0	1	0	2	
0	0	1	1	3	
0	1	0	0	4	
0	1	0	1	5	
0	1	1	0	6	
0	1	1	1	7	
1	0	0	0	8	
1	0	0	1	9	
1	0	1	0	10	
1	0	1	1	11	
1	1	0	0	12	
1	1	0	1	13	
1	1	1	0	14	
1	1	1	1	15	volts

(b)

Fig. 10-2 Conversor D/A de quatro bits com saída em tensão.

EXEMPLO 10-2

Um conversor D/A de cinco bits produz V_{OUT} = 0,2 V para uma entrada digital de 00001. Determine o valor de V_{OUT} para uma entrada de 11111.

Solução

Obviamente, 0,2 V é o peso do LSB. Logo, os pesos dos outros bits devem ser 0,4 V, 0,8 V, 1,6 V e 3,2 V, respectivamente. Então, para uma entrada digital de 11111, o valor de V_{OUT} será 3,2 V + 1,6 V + 0,8 V + 0,4 V + 0,2 V = 6,2 V.

Resolução (Tamanho do Degrau)

A *resolução* de um conversor D/A é definida como a menor alteração que pode ocorrer na saída analógica como resultado de uma mudança na entrada digital. Consultando a tabela na Fig. 10-2, podemos ver que a resolução é 1 V, já que V_{OUT} não pode mudar menos do que 1 V quando o valor da entrada digital é alterado. A resolução é sempre igual ao peso do LSB e também é chamada de **tamanho do degrau**, tendo em vista que é o quanto V_{OUT} muda conforme o valor digital de entrada é alterado de um degrau para o próximo. Isto está mais bem ilustrado na Fig. 10-3, onde as saídas de um contador binário de quatro bits acionam as entradas do conversor D/A. Conforme o contador vai sendo incrementado pelos seus 16 estados pelo sinal de clock, a saída do conversor D/A apresenta uma forma de onda do tipo **escada** que aumenta 1 V por degrau. Quando o contador está em 1111, a saída do conversor D/A está no seu valor máximo de 15 V; é a sua **saída de fundo de escala**. Quando o contador recicla para 0000, a saída do D/A retorna para 0 V. A resolução ou tamanho do degrau é o tamanho dos saltos da forma de onda do tipo escada; neste caso, cada degrau é de 1 V.

Note que a escada tem 16 níveis correspondentes aos 16 estados de entrada, mas existem apenas 15 degraus ou saltos entre o nível de 0 V e o fundo de escala. Em geral, para um conversor D/A de N bits, o número de níveis diferentes será 2^N, e o número de degraus será $2^N - 1$.

Você já deve ter percebido que resolução (tamanho do degrau) é o mesmo que o fator de proporcionalidade na relação de entrada e saída do conversor D/A:

$$\text{saída analógica} = K \times \text{entrada digital}$$

Uma nova interpretação desta expressão seria que a entrada digital é igual ao número de degraus, K é a quantidade de tensão (ou corrente) por degrau, e a saída analógica é o produto dos dois. Temos agora um modo conveniente de calcular o valor de K para o D/A:

$$\text{resolução} = K = \frac{A_{fs}}{(2^n - 1)} \qquad (10\text{-}2)$$

onde A_{fs} é a saída de fundo de escala e n é o número de bits.

EXEMPLO 10-3A

Qual é a resolução (tamanho do degrau) do conversor D/A do Exemplo 10-2? Descreva o sinal de saída do tipo escada deste conversor D/A.

Solução

O LSB para este conversor tem um peso de 0,2 V. Esta é a resolução ou tamanho do degrau. A forma de onda do tipo escada pode ser gerada conectando-se um contador de cinco bits nas entradas do conversor D/A. A escada terá 32 níveis desde 0 V até a saída de fundo de escala de 6,2 V, e 31 degraus de 0,2 V cada.

EXEMPLO 10-3B

Para o conversor D/A do Exemplo 10-2, determine V_{OUT} para uma entrada digital de 10001.

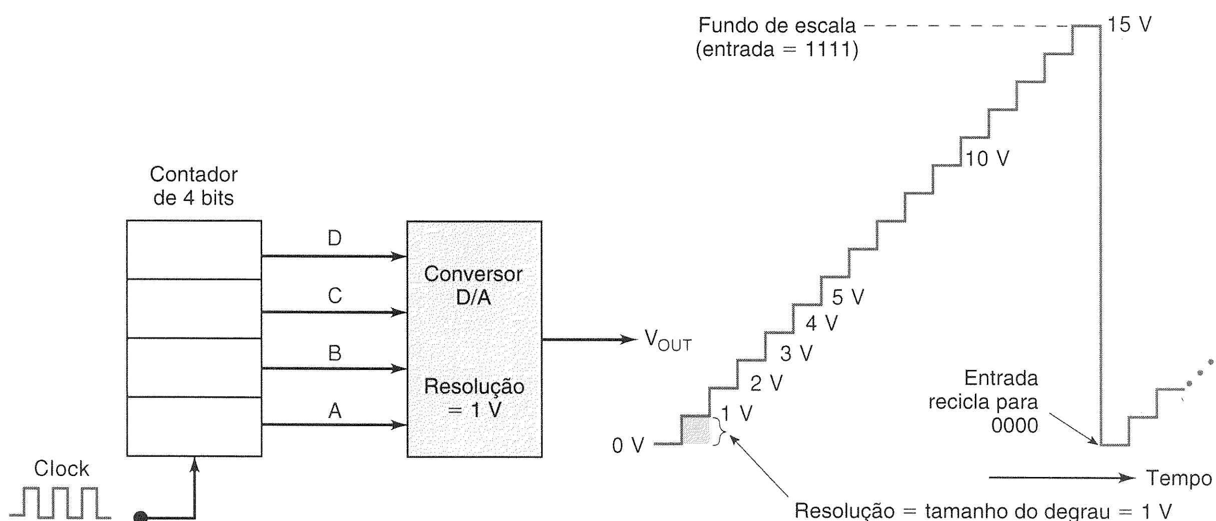


Fig. 10-3 Formas de onda de saída de um conversor D/A com as entradas sendo acionadas por um contador binário.

Solução

O tamanho do degrau é 0,2 V, que é o fator de proporcionalidade *K*. A entrada digital é 10001 = 17_{10}. Logo, temos

$$V_{OUT} = (0,2 \text{ V}) \times 17 = 3,4 \text{ V}$$

Resolução Percentual

Embora a resolução possa ser expressa como a quantidade de tensão ou corrente por degrau, também é útil expressá-la como uma porcentagem da *saída de fundo de escala*. Para ilustrar, o conversor D/A da Fig. 10-3 tem uma saída máxima de fundo de escala de 15 V (quando a entrada digital é 1111). O tamanho do degrau é 1 V, o que resulta numa resolução percentual de

$$\text{resolução\%} = \frac{\text{tamanho de degrau}}{\text{fundo de escala (F.S.)}} \times 100 \qquad (10\text{-}3)$$

$$= \frac{1 \text{ V}}{15 \text{ V}} \times 100\% = 6,67\%$$

EXEMPLO 10-4

Um conversor D/A de 10 bits tem um tamanho de degrau de 10 mV. Determine a tensão de saída de fundo de escala e a resolução percentual.

Solução

Com 10 bits, existirão $2^{10} - 1 = 1023$ degraus de 10 mV cada. A saída de fundo de escala será, portanto, 10 mV × 1023 = 10,23 V, e

$$\text{resolução \%} = \frac{10 \text{ mV}}{10,23 \text{ V}} \times 100\% \approx 0,1\%$$

O Exemplo 10-4 ajuda a ilustrar o fato de que a resolução percentual se torna menor conforme o número de bits de entrada aumenta. Na verdade, a resolução percentual também pode ser calculada por

$$\text{resolução\%} = \frac{1}{\text{número total de degraus}} \times 100\% \qquad (10\text{-}4)$$

Para um código binário de entrada com *N* bits o número total de degraus é $2^N - 1$. Logo, para o exemplo anterior,

$$\text{resolução \%} = \frac{1}{2^{10} - 1} \times 100\%$$

$$= \frac{1}{1023} \times 100\%$$

$$\approx 0,1\%$$

Isto significa que é *somente o número de bits* que determina a resolução *percentual*. Aumentando-se o número de bits, aumenta-se o número de degraus para alcançar o fundo de escala, de modo que cada degrau é uma parte menor da tensão de fundo de escala. A maioria dos fabricantes de conversores D/A especifica resolução como sendo o número de bits.

O que Significa Resolução?

Um conversor D/A não pode produzir uma faixa contínua de valores de saída, e portanto, estritamente falando, sua saída não é verdadeiramente analógica. Um conversor D/A produz um conjunto finito de valores de saída. Em nosso exemplo de temperatura da água da Seção 10-1, o computador gera uma saída digital para fornecer uma tensão analógica entre 0 e 10 V para uma válvula controlada eletricamente. A resolução do conversor D/A (número de bits) determina quantos valores de tensão possíveis o computador pode enviar para a válvula. Se um conversor D/A de seis bits for usado, existirão 63 degraus possíveis de 0,159 V entre 0 e 10 V. Quando um conversor D/A de oito bits for usado, existirão 255 degraus possíveis de 0,039 V entre 0 e 10 V. Quanto maior for o número de bits, mais fina é a resolução (menor tamanho do degrau).

O projetista deve decidir que resolução é necessária com base na performance desejada para o sistema. A resolução limita o quanto a saída de um conversor D/A pode estar próxima de um determinado valor analógico. Geralmente, o custo dos conversores D/A aumenta com o número de bits, e, portanto, o projetista usará somente um D/A com a quantidade de bits que for necessária.

EXEMPLO 10-5

A Fig. 10-4 mostra um computador controlando a velocidade de um motor. A corrente analógica de 0 a 2 mA do conversor D/A é amplificada para produzir velocidades no motor de 0 a 1000 rpm (rotações por minuto). Quantos bits deveriam ser usados se o computador tivesse que ser capaz de produzir uma velocidade no motor que estivesse, no máximo, a 2 rpm da velocidade desejada?

Solução

A velocidade do motor vai variar desde de 0 até 1000 rpm conforme a saída do conversor D/A for de zero até o fundo de escala. Cada degrau na saída do conversor D/A produzirá um degrau na velocidade do motor. Queremos que o tamanho do degrau seja menor do que 2 rpm. Logo, precisamos de pelo menos 500 degraus (1000/2). Agora precisamos determinar quantos bits são necessários para que existam pelo menos 500 degraus de zero até o fundo de escala. Sabemos que o número de degraus é $2^N - 1$, e portanto

$$2^N - 1 \geq 500$$

ou

$$2^N \geq 501$$

Tendo em vista que $2^8 = 256$ e $2^9 = 512$, o menor número de bits que produziria pelo menos 500 degraus é *nove*. Poderíamos utilizar mais do que nove bits, mas isto aumentaria o custo do conversor D/A.

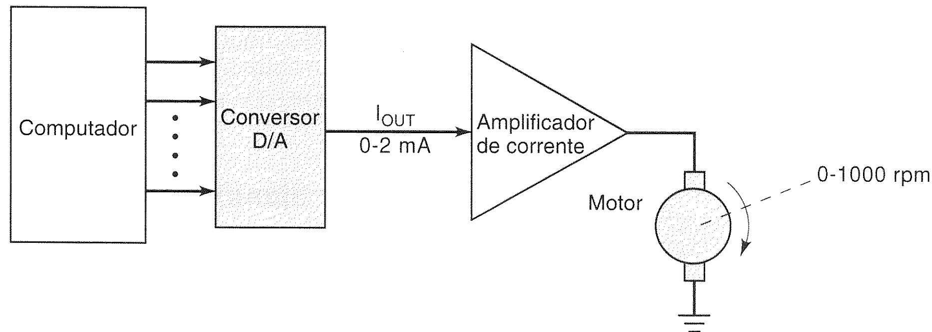


Fig. 10-4 Exemplo 10-5.

EXEMPLO 10-6

Utilizando nove bits, quão próximo de 326 rpm a velocidade do motor pode ser ajustada?

Solução

Com nove bits, existem 511 degraus ($2^9 - 1$). Logo, a velocidade do motor vai aumentar em degraus de 1000 rpm/511 = 1,957 rpm. O número de degraus necessários para alcançar 326 rpm é 326/1,957 = 166,58. Este não é um número inteiro de degraus, e portanto vamos arredondar para 167. A velocidade real do motor no degrau 167 será 167 × 1,957 = 326,8 rpm. Assim, o computador deve enviar o equivalente binário de nove bits de 167_{10} para produzir a velocidade desejada para o motor dentro da resolução do sistema.

Em todos os nossos exemplos, admitimos que os conversores D/A foram perfeitamente precisos na produção da saída analógica que é diretamente proporcional à entrada binária, e que a resolução é a única coisa que limita quão perto podemos chegar de um valor analógico. Isto, naturalmente, não é real, uma vez que qualquer dispositivo contém imprecisões. Examinaremos as causas e efeitos da imprecisão dos conversores D/A posteriormente em uma outra seção.

Código de Entrada BCD

Os conversores D/A que estudamos até agora utilizaram um código binário de entrada. Muitos conversores D/A usam o código de entrada BCD onde grupos de código de quatro bits são utilizados para cada dígito decimal. A Fig. 10-5 mostra o diagrama de um conversor de oito bits (dois dígitos) deste tipo. Cada grupo de código de quatro bits pode variar desde 0000 até 1001, e portanto as entradas BCD representam os números de 00 até 99. Dentro de cada grupo de código os pesos dos diferentes bits estão na proporção binária normal (1, 2, 4, 8), mas os pesos relativos de cada grupo diferem por um fator 10. A Fig. 10-5 mostra os pesos relativos dos vários bits. Note que os bits que constituem o código BCD para o dígito mais significativo (MSD) têm um peso relativo que é 10 vezes o dos bits correspondentes ao LSD.

EXEMPLO 10-7A

Se o peso de A_0 é 0,1 V na Fig. 10-5, determine os seguintes valores:

(a) Tamanho do degrau
(b) Saída de fundo de escala e resolução percentual
(c) V_{OUT} para $D_1C_1B_1A_1 = 0101$ e $D_0C_0B_0A_0 = 1000$

Solução

(a) O tamanho do degrau é o tamanho do LSB do LSD, 0,1 V.
(b) Existem 99 degraus, pois temos dois dígitos BCD. Assim, a saída de fundo de escala é 99 × 0,1 = 9,9 V. A resolução é [utilizando a equação (10-3)]

$$\frac{\text{tamanho do degrau}}{\text{F.S.}} \times 100\% = \frac{0,1}{9,9} \times 100\% \approx 1\%$$

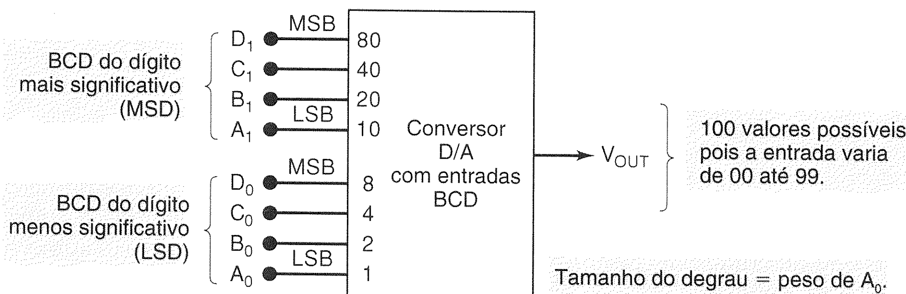


Fig. 10-5 Conversor D/A utilizando código de entrada BCD. Este conversor aceita uma entrada de dois dígitos e gera 100 valores analógicos possíveis para a saída.

Também poderíamos ter usado a equação (10-4) para calcular a resolução percentual, já que o número de degraus é 99.

(c) Os pesos exatos em volts estão listados na Tabela 10-2. Um modo de determinar V_{OUT} para uma certa entrada é somar os pesos de todos os bits que são 1. Assim, para uma entrada de 0101 1000, temos

$$V_{OUT} = \overbrace{4\text{ V}}^{C_1} + \overbrace{1\text{ V}}^{A_1} + \overbrace{0{,}8\text{ V}}^{D_0} = 5{,}8\text{ V}$$

Uma maneira mais fácil é perceber que o código de entrada BCD representa 58_{10} e o tamanho do degrau é 0,1 V, portanto

$$V_{OUT} = (0{,}1\text{ V}) \times 58 = 5{,}8\text{ V}$$

TABELA 10-2

MSD				LSD			
D_1	C_1	B_1	A_1	D_0	C_0	B_0	A_0
8,0	4,0	2,0	1,0	0,8	0,4	0,2	0,1

EXEMPLO 10-7B

Um certo conversor digital-analógico BCD de 12 bits tem uma saída de fundo de escala de 9,99 V.

(a) Determine a resolução percentual.
(b) Determine o tamanho do degrau do conversor.

Solução

(a) Doze bits correspondem a três dígitos decimais, isto é, números decimais desde 000 até 999. Portanto, a saída deste conversor D/A possui 999 degraus possíveis de 0 a 9,99 V. Logo, temos

$$\text{resolução \%} = \frac{1}{\text{número de degraus}} \times 100\%$$

$$= \frac{1}{999} \times 100\%$$

$$\approx \mathbf{0{,}1\%}$$

(b) $$\text{tamanho do degrau} = \frac{\text{F.S.}}{\text{número de degraus}}$$

$$= 9{,}99\text{ V}/999$$
$$= \mathbf{0{,}01\text{ V}}$$

Conversores D/A Bipolares

Até este ponto, consideramos que a entrada binária para um conversor D/A era um número não-sinalizado e que a saída do conversor D/A era uma tensão ou corrente positiva. Alguns conversores D/A são projetados para produzir valores positivos ou negativos, tais como −10 V a 10 V. Isto geralmente é feito utilizando-se a entrada binária como um número sinalizado com o MSB sendo o bit de sinal (0 para + e 1 para −). Valores negativos de entrada freqüentemente são representados na forma de complemento a 2, embora a forma de sinal e magnitude também seja usada por alguns conversores D/A. Por exemplo, suponha que temos um conversor D/A bipolar de seis bits que utiliza o sistema de complemento a 2 e tem resolução de 0,2 V. Os valores binários de entrada variam de 100000 (−32) até 011111 (+31) para produzir saídas analógicas na faixa de −6,4 V até +6,2 V. Existem 63 degraus ($2^6 - 1$) de 0,2 V entre os limites negativo e positivo.

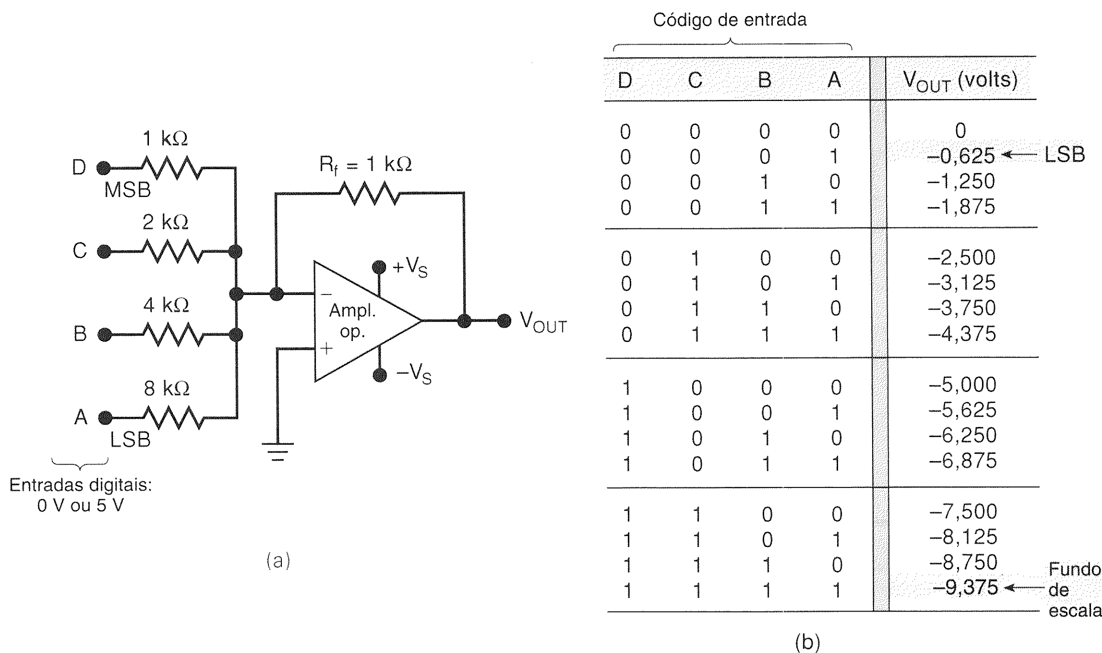
Questões de Revisão

1. Um conversor D/A de oito bits tem uma saída de 3,92 mA para uma entrada de 01100010. Quais são a resolução deste conversor D/A e sua saída de fundo de escala?
2. Qual é o peso do MSB do conversor D/A da questão 1?
3. Qual é a resolução percentual de um conversor D/A de oito bits?
4. Quantas tensões de saída diferentes um conversor D/A de 12 bits pode produzir?
5. Para o sistema da Fig. 10-4, quantos bits deveriam ser usados se o computador tivesse que controlar a velocidade do motor para mantê-la, no máximo, a 4 rpm da velocidade desejada?
6. Considere um conversor D/A de 12 bits com entradas BCD e uma resolução de 10 mV. Qual será sua saída para uma entrada de 100001110011?
7. *Verdadeiro ou falso:* A resolução percentual de um conversor D/A depende *apenas* do número de bits.
8. Qual é a vantagem de uma resolução mais fina?

10-3 CIRCUITOS CONVERSORES D/A

Existem vários métodos e circuitos para implementar a operação D/A que foi descrita. Examinaremos diversos esquemas básicos para adquirir uma boa compreensão das idéias utilizadas. Não é importante se familiarizar com todos os diversos esquemas de circuitos, pois conversores D/A estão disponíveis como CIs ou como módulos encapsulados que não requerem qualquer conhecimento dos circuitos. Por outro lado, é importante saber as características significativas de performance dos conversores D/A, em geral, de modo que eles possam ser utilizados inteligentemente. Isto será estudado na Seção 10-4.

A Fig. 10-6(a) mostra o circuito básico para um tipo de conversor D/A de quatro bits. As entradas *A*, *B*, *C* e *D* são entradas binárias que admitimos terem valores de 0 ou 5 V. O *amplificador operacional* é empregado como um amplificador somador, que produz a soma ponderada destas tensões de entrada. Devemos lembrar que o amplificador somador multiplica cada tensão de entrada pela razão do resistor de realimentação R_F com seu correspondente resistor de entrada R_{IN}. Neste circuito $R_F = 1\text{ k}\Omega$, e o resistor de entrada varia de 1 até 8 kΩ. A entrada *D* tem $R_{IN} = 1\text{ k}\Omega$,



D	C	B	A	V_{OUT} (volts)
0	0	0	0	0
0	0	0	1	−0,625 ← LSB
0	0	1	0	−1,250
0	0	1	1	−1,875
0	1	0	0	−2,500
0	1	0	1	−3,125
0	1	1	0	−3,750
0	1	1	1	−4,375
1	0	0	0	−5,000
1	0	0	1	−5,625
1	0	1	0	−6,250
1	0	1	1	−6,875
1	1	0	0	−7,500
1	1	0	1	−8,125
1	1	1	0	−8,750
1	1	1	1	−9,375 ← Fundo de escala

Fig. 10-6 Conversor D/A simples, que utiliza um amplificador operacional na configuração somador com resistores ponderados.

portanto o amplificador somador passa a tensão em D sem nenhuma atenuação. A entrada C tem $R_{IN} = 2\ k\Omega$, portanto ela será atenuada de 1/2. Analogamente, a entrada B será atenuada de 1/4, e a entrada A por 1/8. Logo, a saída do amplificador pode ser expressa como

$$V_{OUT} = -(V_D + \tfrac{1}{2}V_C + \tfrac{1}{4}V_B + \tfrac{1}{8}V_A) \qquad (10\text{-}5)$$

O sinal negativo está presente porque o amplificador somador é um amplificador inversor de polaridade, mas isto não nos importa agora.

É claro que a saída do amplificador somador é uma tensão analógica que representa a soma ponderada das entradas digitais, conforme mostrado na tabela na Fig. 10-6(b). Esta tabela relaciona todas as possíveis combinações de entrada e a tensão de saída resultante do amplificador. A saída é avaliada para qualquer condição de entrada, colocando-se as entradas apropriadas em 0 ou 5 V. Por exemplo, se a entrada digital é 1010, então $V_D = V_B = 5$ V e $V_C = V_A = 0$ V. Logo, usando a equação (10-5),

$$\begin{aligned} V_{OUT} &= -(5\text{ V} + 0\text{ V} + \tfrac{1}{4} \times 5\text{ V} + 0\text{ V}) \\ &= -6{,}25\text{ V} \end{aligned}$$

A resolução deste conversor D/A é igual ao peso do LSB, que é $1/8 \times 5$ V = 0,625 V. Como mostrado na tabela, a saída analógica aumenta de 0,625 V conforme o número binário de entrada aumenta.

EXEMPLO 10-8

(a) Determine o peso de cada bit de entrada da Fig. 10-6(a).

(b) Substitua R_F por 250 Ω e determine a saída de fundo de escala.

Solução

(a) O MSB passa com ganho = 1, logo seu peso na saída é 5 V. Assim,

MSB → 5 V
Segundo MSB → 2,5 V
Terceiro MSB → 1,25 V
Quarto MSB = LSB → 0,625 V

(b) Se R_F é reduzido por uma fator 4, para 250 Ω, cada peso de entrada será quatro vezes *menor* do que os valores originais. Assim, a saída de fundo de escala será reduzida pelo mesmo fator e se tornará $-9{,}375/4 = -2{,}344$ V.

Se analisarmos os valores dos resistores de entrada na Fig. 10-6, não deveríamos ficar surpresos por eles serem *binariamente ponderados*. Em outras palavras, iniciando-se com o resistor do MSB, os valores dos resistores aumentam por um fator de 2. Isto, naturalmente, produz a desejada ponderação na tensão de saída.

Precisão da Conversão

A tabela na Fig. 10-6(b) fornece os valores *ideais* de V_{OUT} para as diversas entradas. O quão perto o circuito consegue produzir estes valores depende principalmente de dois fatores: (1) a precisão dos resistores de entrada e de realimentação e (2) a precisão dos níveis de tensão das entradas. Os resistores podem ser construídos muito precisos (dentro de 0,01 por cento dos valores desejados), mas os níveis de tensão das entradas devem ser tratados de modo diferente. Deve ficar claro que as entradas digitais não podem ser tomadas diretamente das saídas de

FFs ou portas lógicas porque os níveis lógicos de saída destes dispositivos não são valores precisos como 0 V e 5 V, mas variam em determinadas faixas. Por esta razão, é necessário adicionar alguns circuitos entre a entrada digital e seu resistor do amplificador somador, conforme mostrado na Fig. 10-7.

Cada entrada digital controla uma chave semicondutora, tal como a porta de transmissão CMOS que estudamos no Cap. 8. Quando a entrada está em ALTO, a chave fecha e conecta uma *fonte de precisão de referência* ao resistor de entrada; quando a entrada está em BAIXO, a chave está aberta. A fonte de referência produz uma tensão muito estável e precisa necessária para gerar uma saída analógica precisa.

Conversor D/A com Saída em Corrente

A Fig. 10-8(a) mostra um esquema básico para gerar uma corrente analógica de saída proporcional à entrada binária. O circuito mostrado é um conversor D/A de quatro bits que utiliza resistores ponderados binariamente. O circuito usa quatro ramos paralelos para a corrente, cada um controlado por uma chave semicondutora, tal como a porta de transmissão CMOS. O estado de cada chave é controlado pelo nível lógico da entrada binária. A corrente em cada ramo é determinada por uma tensão precisa de referência, V_{REF}, e um resistor de precisão. Os resistores são ponderados binariamente, de modo que as correntes serão ponderadas binariamente, e a corrente total, I_{OUT}, será a soma das correntes individuais. O caminho do MSB possui o menor resistor, R; o caminho do lado tem um resistor com o dobro deste valor; e assim por diante. A corrente de saída pode ser levada a fluir através de uma carga R_L, que é muito menor do que R, de modo que não afeta o valor da corrente. Idealmente, R_L deveria ser um curto para a terra.

EXEMPLO 10-9

Admita que V_{REF} = 10 V e R = 10 kΩ. Determine a resolução e a saída de fundo de escala deste conversor D/A. Considere que R_L é muito menor do que R.

Solução

$I_{OUT} = V_{REF}/R = 1$ mA. Este é o peso do MSB. As outras três correntes serão 0,5, 0,25 e 0,125 mA. O LSB é 0,125 mA, que também é a resolução.

A saída de fundo de escala ocorrerá quando as entradas binárias estiverem todas em ALTO, de modo que cada chave de corrente esteja fechada e

$$I_{OUT} = 1 + 0{,}5 + 0{,}25 + 0{,}125 = 1{,}875 \text{ mA}$$

Note que a corrente de saída é proporcional a V_{REF}. Se V_{REF} for aumentado ou diminuído, a resolução e a saída de fundo de escala mudarão proporcionalmente.

Para I_{OUT} ser precisa, R_L deveria ser um curto para a terra. Uma maneira comum de implementar isto é utilizar um conversor de corrente-tensão baseado em amplificador operacional, como mostrado na Fig. 10-8(b). A corrente I_{OUT} do conversor D/A está conectada à entrada "–" do amplificador operacional, que está praticamente na terra. A realimentação negativa no amplificador operacional força uma corrente igual a

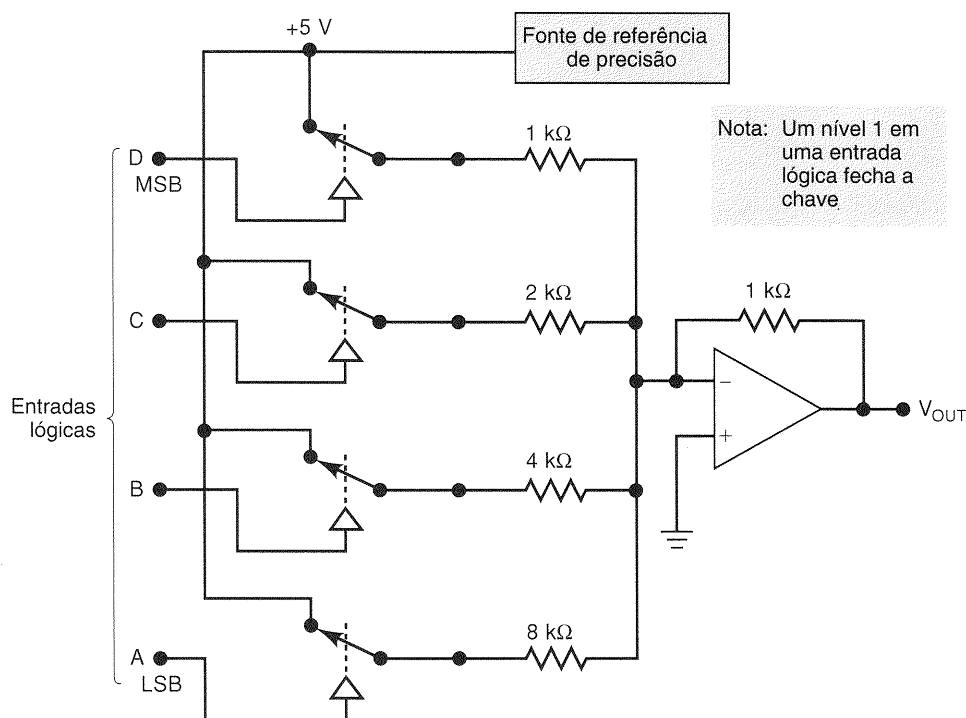


Fig. 10-7 Conversor D/A de quatro bits completo, incluindo a fonte de precisão de referência.

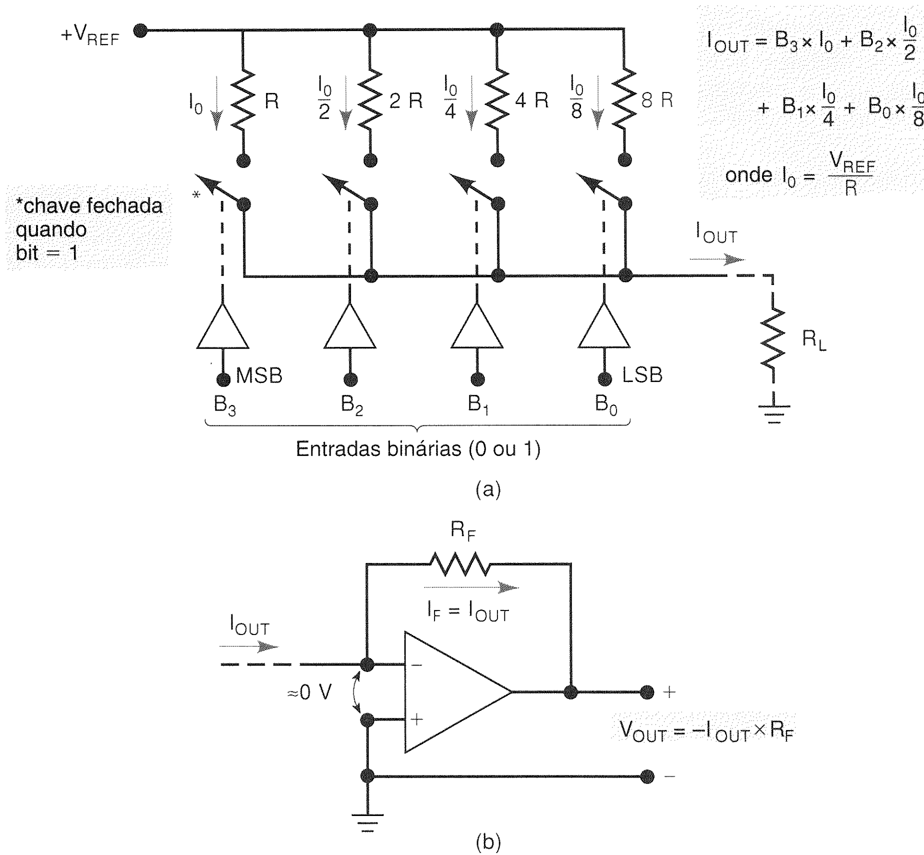


Fig. 10-8 (a) Conversor D/A básico com saída em corrente; (b) conexão a um conversor corrente-tensão baseado em amplificador operacional.

I_{OUT} a fluir através de R_F para produzir $V_{OUT} = -I_{OUT} \times R_F$. Assim, V_{OUT} será uma tensão analógica proporcional à entrada binária do conversor D/A. Esta saída analógica pode acionar uma ampla faixa de cargas sem ficar sobrecarregada.

Rede *R*/2*R*

Os circuitos de conversores D/A que apresentamos até agora utilizaram resistores ponderados para produzir o peso apropriado de cada bit. Apesar de este método funcionar na teoria, ele tem algumas limitações práticas. O maior problema é a grande diferença entre os valores dos resistores do LSB e do MSB, especialmente em conversores D/A de alta resolução (isto é, de muitos bits). Por exemplo, se o resistor do MSB é de 1 kΩ num conversor D/A de 12 bits, o resistor do LSB será superior a 2 MΩ. Com a tecnologia atual de fabricação de CIs, é muito difícil produzir valores de resistências por uma faixa tão ampla que ainda mantenham uma razão precisa, especialmente com as variações de temperatura.

Por isto é preferível ter um circuito que utiliza resistências que tenham valores próximos. Um dos circuitos conversores D/A mais comumente usados que satisfazem este requisito é a *rede R/2R*, onde os valores de resistência estão numa faixa de 2 para 1. Um conversor D/A deste tipo é mostrado na Fig. 10-9.

Note como os resistores estão arranjados, e repare, especialmente, que somente dois valores diferentes são utilizados, *R* e 2*R*. A corrente I_{OUT} depende da posição das quatro chaves, e as entradas binárias $B_3B_2B_1B_0$ controlam os estados das chaves. Esta corrente pode fluir através do conversor de corrente-tensão baseado em amplificador operacional para gerar V_{OUT}. Não vamos analisar detalhadamente este circuito aqui, mas pode ser mostrado que o valor de V_{OUT} é dado pela seguinte expressão

$$V_{OUT} = \frac{-V_{REF}}{8} \times B \qquad (10\text{-}6)$$

onde *B* é o valor da entrada binária, que pode variar de 0000 (0) até 1111 (15).

EXEMPLO 10-10

Considere que $V_{REF} = 5$ V para o conversor D/A na Fig. 10-9. Quais são a resolução e a saída de fundo de escala deste conversor?

Solução

A resolução é igual ao peso do LSB, o qual podemos determinar fazendo $B = 0001 = 1$ na equação (10-6):

$$\text{resolução} = \frac{-5\text{ V} \times 1}{8}$$

$$= -0{,}625\text{ V}$$

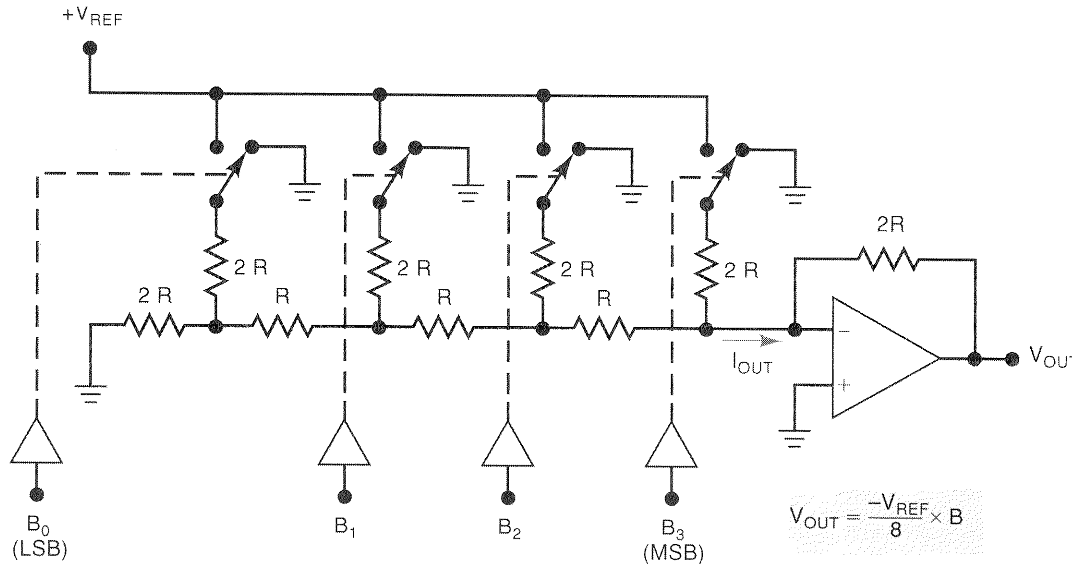


Fig. 10-9 Conversor D/A básico do tipo *R*/2*R*.

A saída de fundo de escala ocorre para $B = 1111 = 15_{10}$. Novamente usando a equação (10-6),

$$\text{fundo de escala} = \frac{-5\ \text{V} \times 15}{8}$$

$$= -9{,}375\ \text{V}$$

Questões de Revisão

1. Qual é a vantagem dos conversores D/A do tipo *R*/2*R* sobre aqueles que usam resistores ponderados?
2. Um certo conversor D/A de seis bits utiliza resistores binariamente ponderados. Se o resistor do MSB é 20 kΩ, qual é o resistor do LSB?
3. Qual será a resolução se o valor de R_F na Fig. 10-6 for alterado para 800 Ω?
4. O que acontecerá com a resolução e a saída de fundo de escala quando V_{REF} é aumentado de 20%?

10-4 ESPECIFICAÇÕES DE CONVERSORES D/A

Atualmente, uma ampla variedade de conversores D/A está disponível como CI ou como módulos encapsulados. Devemos estar familiarizados com as especificações mais importantes dos fabricantes de modo a avaliar um conversor D/A para uma aplicação em particular.

Resolução

Conforme mencionado anteriormente, a resolução percentual de um conversor D/A depende apenas do número de bits. Por esta razão, os fabricantes usualmente especificam a resolução de um conversor D/A como o número de bits. Um conversor D/A de 10 bits tem uma resolução mais fina do que um de 8 bits.

Precisão

Os fabricantes de conversores D/A têm vários modos de especificar a precisão. Os dois mais comuns são chamados **erro de fundo de escala** e **erro de linearidade**, que normalmente são expressos como uma porcentagem da saída de fundo de escala (% F.S.).

O erro de fundo de escala é o máximo desvio da saída do conversor D/A em relação ao valor esperado (ideal), expresso como uma porcentagem do fundo de escala. Por exemplo, admita que o conversor D/A da Fig. 10-6 tem uma precisão de ±0,01% F.S. Como este conversor possui um fundo de escala de 9,375 V, esta porcentagem resulta em

$$\pm 0{,}01\% \times 9{,}375\ \text{V} = \pm 0{,}9375\ \text{mV}$$

Isto significa que a saída deste conversor D/A pode, a qualquer momento, estar a até 0,9375 mV do seu valor esperado.

O erro de linearidade é o máximo desvio em tamanho do degrau do tamanho de degrau ideal. Por exemplo, o conversor D/A da Fig. 10-6 tem um tamanho de degrau esperado de 0,625 V. Se este conversor tiver um erro de linearidade de ±0,01% F.S., significaria que o *tamanho de degrau* real poderia estar a até 0,9375 mV do esperado.

É importante compreender que a precisão e a resolução de um conversor D/A devem ser compatíveis. É ilógico ter uma resolução de, digamos, 1 por cento e uma precisão de 0,1 por cento, ou vice-versa. Para ilustrar, um conversor D/A com uma resolução de 1% e uma saída de F.S. de 10 V pode produzir uma tensão de saída analógica a 0,1 V de qualquer valor desejado, admitindo-se precisão perfeita. Não faz sentido ter uma precisão de 0,01% F.S. (ou 1mV), que custa caro, se a resolução já limita a proximidade do valor desejado para 0,1 V. O mesmo pode ser dito para uma re-

solução que é muito fina (muitos bits) enquanto a precisão é pobre; é um desperdício de bits de entrada.

EXEMPLO 10-11

Um determinado conversor D/A de oito bits tem uma saída de fundo de escala de 2 mA e um erro de fundo de escala de ±0,5% F.S. Qual é a faixa de valores de saída possíveis para uma entrada de 10000000?

Solução

O tamanho do degrau é 2 mA/255 = 7,84 μA. Tendo em vista que 10000000 = 128_{10}, a saída ideal seria 128 × 7,84 μA = 1004 μA. O erro poderia ser de até

$$\pm 0,5\% \times 2 \text{ mA} = \pm 10 \ \mu\text{A}$$

Logo, a saída real pode desviar deste valor a partir dos 1004 μA ideais, portanto a saída real pode ter qualquer valor entre 994 e 1014 μA.

Erro de Offset

Idealmente, a saída de um conversor D/A estará com zero volts quando a entrada binária é formada apenas por 0s. Na prática, entretanto, existe uma tensão de saída muito pequena nesta situação; isto é chamado de **erro de offset**. Este erro de offset, se não for corrigido, será adicionado às saídas esperadas do conversor D/A em *todos* os casos de entrada. Por exemplo, digamos que um conversor D/A de quatro bits apresente um erro de offset de +2 mV e um tamanho *perfeito* de degrau de 100 mV. A Tabela 10-3 mostra as saídas ideais e reais do conversor D/A para diversos casos de entrada. Note que a saída real é 2 mV maior do que a esperada; isto é devido ao erro de offset. O erro de offset pode ser negativo ou positivo.

TABELA 10-3

Código de Entrada	Saída Ideal (mV)	Saída Real (mV)
0000	0	2
0001	100	102
1000	800	802
1111	1500	1502

Muitos conversores D/A possuem um ajuste externo de offset que nos permite zerar o offset. Isto usualmente é implementado aplicando-se 0s na entrada do conversor D/A e monitorando a saída, enquanto um *potenciômetro de ajuste de offset* é ajustado até que a saída esteja tão próxima de 0 V quanto for necessário.

Tempo de Estabilização

Usualmente, a velocidade de operação de um conversor D/A é especificada fornecendo-se o seu **tempo de estabilização**, que é o tempo necessário para a saída do conversor D/A ir de zero até o fundo de escala quando a entrada binária muda de todos os bits em 0 para todos em 1. Na verdade, o tempo de estabilização é medido como o tempo para a saída do conversor D/A se estabilizar dentro de ±1/2 tamanho do degrau (resolução) do seu valor final. Por exemplo, se um conversor D/A tem uma resolução de 10 mV, o tempo de estabilização é medido como o tempo que a saída leva para se estabilizar dentro de 5 mV do seu valor de fundo de escala.

Valores típicos para tempos de estabilização variam de 50 ns até 10 μs. Em geral, conversores D/A com saída em corrente possuem tempos de estabilização mais curtos do que aqueles com saída em tensão. Por exemplo, o DAC1280 pode operar tanto com saída em corrente como com saída em tensão. No modo de saída em corrente, seu tempo de estabilização é 300 ns; no modo de saída em tensão, seu tempo de estabilização é 2,5 μs. O principal motivo para esta diferença é o tempo de resposta do amplificador operacional que é usado como um conversor de corrente-tensão.

Monotonicidade

Um conversor D/A é **monotônico** se sua saída aumenta conforme a entrada binária é incrementada de um valor para o próximo. Um outro modo de descrever isto é que a saída do tipo escada não terá nenhum degrau para baixo conforme a entrada binária for incrementada de zero até o fundo de escala.

Questões de Revisão

1. Defina *erro de fundo de escala.*
2. O que é tempo de estabilização?
3. Descreva erro de offset e seus efeitos sobre a saída de um conversor D/A.
4. Por que os conversores D/A com saída em tensão geralmente são mais lentos do que os de saída em corrente?

10-5 UM CIRCUITO INTEGRADO DE CONVERSOR D/A

O AD7524, um CI CMOS disponível de diversos fabricantes de CIs, é um conversor D/A de oito bits que utiliza uma rede *R*/2*R*. Seu símbolo é apresentado na Fig. 10-10(a). Este conversor D/A tem uma entrada de oito bits que pode ser armazenada internamente sob o controle das entradas de Seleção do Chip (*Chip Select*— $\overline{CS}$) e de ESCRITA (*WRITE*— $\overline{WR}$). Quando estas duas entradas de controle estão em BAIXO, as entradas digitais de dados, $D_7 - D_0$, produzem uma corrente de saída analógica *OUT 1* (o pino *OUT 2* normalmente é aterrado). Quando uma das entradas de controle vai para ALTO, o dado digital de entrada é armazenado, e a saída analógica permanece no nível correspondente ao dado digital armazenado. Neste estado de armazenamento, alterações subseqüentes na entrada digital não têm nenhum efeito sobre *OUT 1*.

O tempo máximo de estabilização para o AD7524 é tipicamente de 100 ns, e sua precisão de fundo de escala é ±0,2

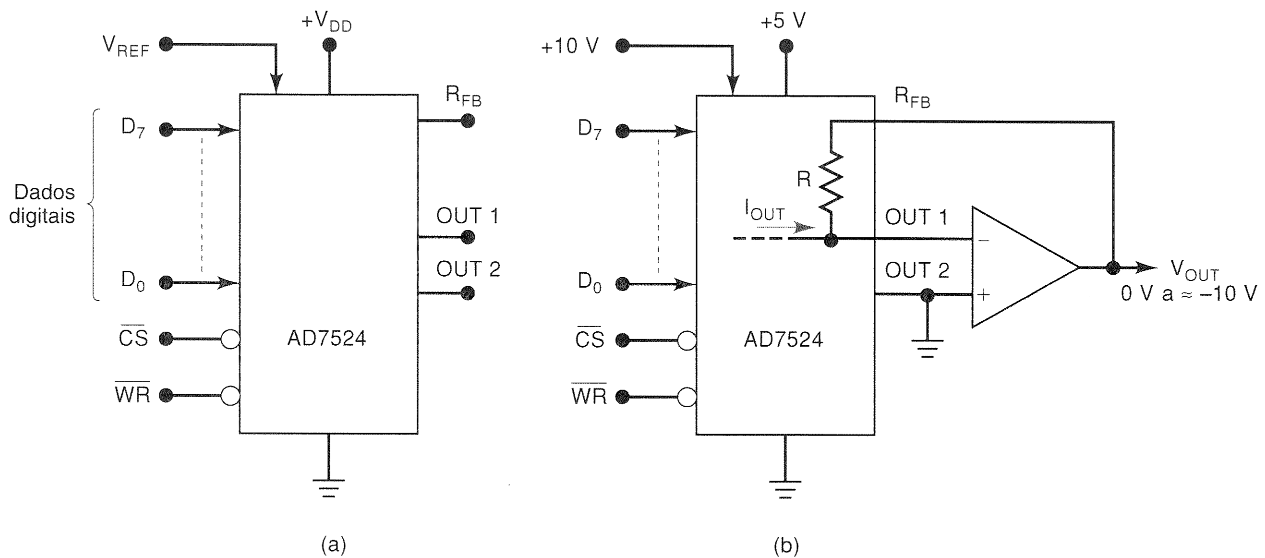


Fig. 10-10 (a) Conversor D/A de oito bits AD7524 com entradas de armazenamento; (b) AD7524 conectado para produzir uma tensão analógica na saída variando de 0 a aproximadamente −10 V.

F.S. O V_{REF} pode variar sobre tensões negativas ou positivas de 0 a 25 V, de modo que correntes analógicas de saída de ambas as polaridades podem ser produzidas. A corrente de saída pode ser convertida em uma tensão utilizando-se um amplificador operacional conectado como na Fig. 10-10(b). Note que o resistor de realimentação do amplificador operacional já está no chip do conversor D/A.

10-6 APLICAÇÕES DE CONVERSORES D/A

Conversores D/A são usados sempre que a saída de um circuito digital deve fornecer uma tensão ou corrente analógica para acionar um dispositivo analógico. Algumas das aplicações mais comuns estão descritas nos parágrafos seguintes.

Controle

A saída digital de um computador pode ser convertida em um sinal de controle analógico para ajustar a velocidade de um motor ou a temperatura de um forno, ou controlar quase qualquer variável física.

Teste Automático

Computadores podem ser programados para gerar os sinais analógicos (através de um conversor D/A) necessários aos testes de circuitos analógicos. A resposta analógica da saída do circuito sob teste normalmente será convertida para um valor digital por um conversor A/D e levada para o computador para ser armazenada, apresentada e algumas vezes analisada.

Reconstrução de Sinais

Em muitas aplicações, um sinal analógico é **digitalizado**; isto é, pontos sucessivos do sinal são convertidos para seus equivalentes digitais e armazenados em memória. Esta conversão é realizada por um conversor analógico-digital (conversor A/D). Um conversor D/A então pode ser usado para converter os dados digitais armazenados de volta para o mundo analógico — um ponto de cada vez —, reconstruindo desta forma o sinal original. Esta combinação de digitalização e reconstrução é utilizada em osciloscópios de memória digital, sistemas de áudio com CDs e em gravação digital de áudio e vídeo. Vamos abordar isto após aprendermos sobre conversores A/D.

Conversão A/D

Muitos tipos de conversores A/D utilizam conversores D/A como parte de seus circuitos, conforme estudaremos na Seção 10-8.

Conversores D/A Seriais

Muitas destas aplicações de conversores D/A envolvem um microprocessador. O problema principal em usar conversores D/A com dados paralelos, que descrevemos até agora, é que eles ocupam muitos bits das portas do microcomputador. Em casos em que a velocidade de transferência de dados não é muito importante, um microprocessador pode fornecer o valor digital para o conversor D/A através de uma interface serial. Conversores D/A seriais estão prontamente disponíveis com um registrador de deslocamento embutido do tipo entrada serial/saída paralela. Muitos destes dispositivos possuem mais do que um conversor no mesmo chip. Os dados digitais, em conjunto com um código que especifica qual dos conversores D/A se deseja, são enviados para o chip, um bit de cada vez. Conforme cada bit chega na entrada do conversor D/A, um pulso é aplicado na entrada de clock serial para deslocar o bit para dentro. Depois do número apropriado de pulsos de clock, o dado é armazenado e convertido para seu valor analógico.

10-7 PESQUISA DE FALHAS EM CONVERSORES D/A

Conversores D/A são tanto digitais quanto analógicos. Pontas de prova lógica e pulsadores podem ser usados nas entradas digitais, mas um medidor ou um osciloscópio deve ser utilizado na saída analógica. Basicamente, existem dois modos de testar a operação de um conversor D/A: um *teste de precisão estático* e um *teste do tipo escada*.

O teste estático se resume a configurar a entrada binária com um valor fixo e medir a saída analógica com um medidor de alta precisão. Este teste é usado para verificar que a saída se encontra dentro da faixa esperada, de acordo com as especificações de precisão do conversor D/A. Se não estiver, podem existir muitas causas possíveis. Algumas delas são:

- Variação nos valores dos componentes internos do conversor D/A (por exemplo, valores de resistores) causada por temperatura, envelhecimento, ou alguns outros fatores. Isto pode facilmente provocar valores de saída fora da faixa de precisão esperada.
- Conexões abertas ou em curto-circuito em qualquer das entradas binárias. Isto poderia impedir que uma entrada tivesse seu peso somado na saída analógica ou fazer com que seu peso estivesse sempre presente na saída. É especialmente difícil de detectar quando a falha está numa das entradas menos significativas.
- Uma tensão de referência com problemas. Como a saída analógica é diretamente dependente de V_{REF}, isto poderia provocar resultados imprevisíveis. Para conversores D/A que utilizam fontes de referência externas, a tensão de referência pode ser facilmente verificada com um voltímetro digital. Muitos conversores D/A possuem tensões de referência internas que não podem ser verificadas, exceto em alguns conversores que disponibilizam a tensão de referência em um pino do CI.
- Erro de offset excessivo causado por envelhecimento dos componentes ou pela temperatura. Isto pode produzir saídas que diferem por um valor fixo. Se o conversor D/A possuir um ajuste de offset externo, este tipo de erro pode inicialmente ser anulado, mas mudanças na temperatura de operação podem causar o reaparecimento do erro de offset.

O teste do tipo escada é usado para verificar a monotonicidade do conversor D/A; isto é, ele verifica se a saída aumenta degrau a degrau conforme a entrada binária vai sendo incrementada, como na Fig. 10-3. Os degraus da escada devem ter o mesmo tamanho, e não deve haver degraus faltando e nem degraus para baixo até que o fundo de escala seja atingido. Este teste pode ajudar a detectar falhas internas e externas que provocam que uma entrada não tenha contribuição ou tenha contribuição permanente para a saída analógica. O exemplo seguinte vai ilustrar isto.

EXEMPLO 10-12

Como seria a forma de onda do tipo escada se a entrada *C* do conversor D/A da Fig. 10-3 estivesse aberta? Admita que as entradas do conversor D/A são compatíveis com TTL.

Solução

Uma conexão aberta em *C* seria interpretada pelo conversor D/A como um nível lógico 1 constante. Assim, isto contribuirá com constantes 4 V para a saída do conversor D/A, de modo que sua forma de onda de saída aparecerá como mostrada na Fig. 10-11. As linhas tracejadas representam a escada original que apareceria se o conversor D/A estivesse funcionando bem. Note que a forma de onda de saída com problemas coincide com a correta durante os instantes em que a entrada *C* estaria normalmente em ALTO.

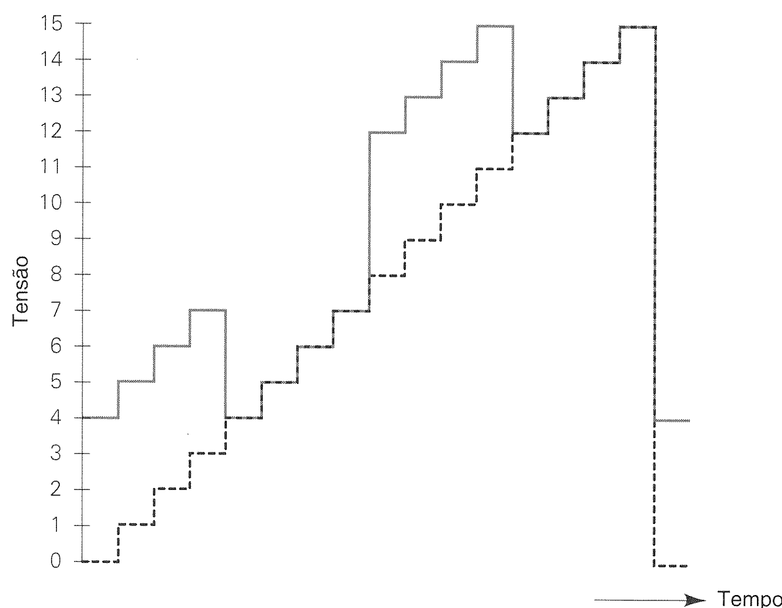


Fig. 10-11 Exemplo 10-12.

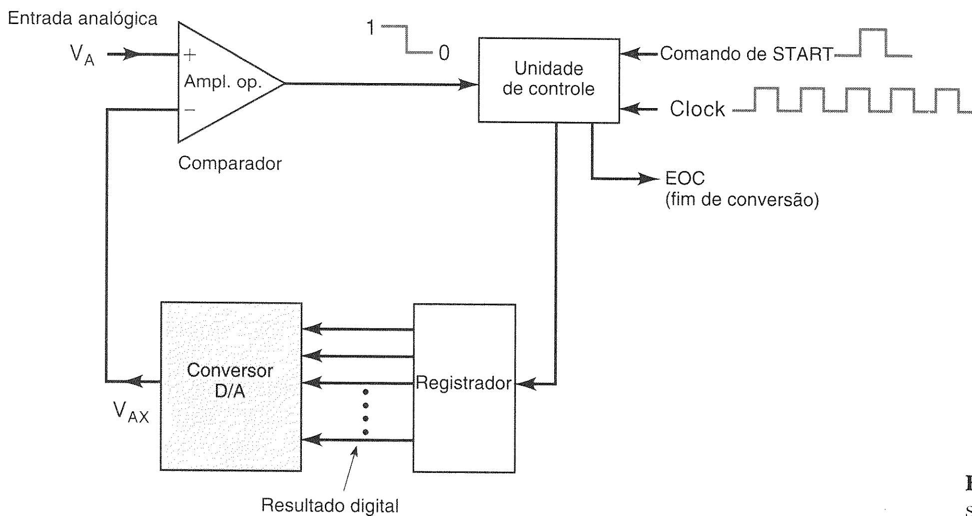


Fig. 10-12 Diagrama geral de uma classe de conversores A/D.

10-8 CONVERSÃO ANALÓGICO-DIGITAL (A/D)

Um **conversor analógico-digital** recebe uma tensão analógica de entrada e depois de um certo tempo produz um código digital de saída que representa a entrada analógica. O processo de conversão A/D geralmente é mais complexo e consome mais tempo do que o processo D/A, e muitos métodos diferentes foram desenvolvidos e utilizados. Examinaremos diversos métodos em detalhe, muito embora possa não ser necessário projetar ou construir conversores A/D (eles são disponíveis como unidades completamente encapsuladas). Entretanto, as técnicas que são usadas fornecem uma compreensão dos fatores que determinam a performance de um conversor A/D.

Muitos tipos importantes de conversores A/D utilizam um conversor D/A como parte de seus circuitos. A Fig. 10-12 é um diagrama de blocos geral para esta classe de conversor A/D. A temporização para a operação é fornecida pelo sinal de clock de entrada. A unidade de controle contém os circuitos lógicos para gerar a seqüência apropriada de operações em resposta ao comando de START, que inicia o processo de conversão. O amplificador operacional comparador tem duas entradas *analógicas* e uma saída *digital* que muda de estado, dependendo de qual das entradas analógicas é maior.

A operação básica dos conversores A/D deste tipo consiste nos seguintes passos:

1. O pulso START inicia a operação.
2. Numa taxa determinada pelo clock, a unidade de controle modifica continuamente o número binário que está armazenado no registrador.
3. O número binário no registrador é convertido para uma tensão analógica, V_{AX}, pelo conversor D/A.
4. O comparador compara V_{AX} com a entrada analógica V_A. Enquanto $V_{AX} < V_A$, a saída do comparador fica em ALTO. Quando V_{AX} excede V_A pelo menos por uma quantidade igual a V_T (tensão de limiar), a saída do comparador vai para BAIXO e pára o processo de modificar o número do registrador. Neste ponto, V_{AX} é uma boa aproximação para V_A. O número digital no registrador, que é o digital equivalente de V_{AX}, também é o equivalente digital aproximado de V_A, dentro da resolução e da precisão do sistema.
5. A lógica de controle ativa o sinal de fim de conversão, *EOC*, quando a conversão está completa.

As diversas variações deste esquema de conversão A/D diferem principalmente na maneira pela qual a seção de controle modifica continuamente os números no registrador. Fora isso, a idéia básica é a mesma, com o registrador mantendo a saída digital desejada quando o processo de conversão está completo.

Questões de Revisão

1. Qual é a função do comparador neste conversor A/D?
2. Onde está o equivalente digital aproximado de V_A quando a conversão está completa?
3. Qual é a função do sinal *EOC*?

10-9 CONVERSOR A/D DE RAMPA DIGITAL

Uma das versões mais simples do conversor A/D geral da Fig. 10-12 utiliza um contador binário como o registrador e permite que o clock incremente o contador um passo de cada vez até que $V_{AX} \geq V_A$. É o chamado **conversor A/D de rampa digital** porque a forma de onda de V_{AX} é uma rampa passo a passo (na verdade uma escada) como aquela mostrada na Fig. 10-3. Também é conhecido como *conversor* A/D *tipo contador*.

A Fig. 10-13 é o diagrama para um conversor A/D de rampa digital. Ele contém um contador, um conversor D/A, um comparador analógico e uma porta AND de controle. A saída do comparador serve como o sinal de fim de conversão ativo em BAIXO, $\overline{EOC}$. Se admitirmos que V_A, a tensão analógica a ser convertida, é positiva, a operação se processa como segue:

1. Um pulso de START é aplicado para ressetar o contador para 0. O nível ALTO em START também inibe os pulsos de clock de passarem pela porta AND para o contador.

2. Com todas as suas entradas em 0, a saída do conversor D/A será $V_{AX} = 0$ V.
3. Como $V_A > V_{AX}$, a saída do comparador, $\overline{EOC}$, estará em ALTO.
4. Quando START retorna para BAIXO, a porta AND é habilitada e os pulsos de clock vão para o contador.
5. Conforme o contador avança, a saída do conversor D/A, V_{AX}, aumenta um degrau por vez, como mostrado na Fig. 10-13(b).
6. Isto continua até que V_{AX} alcança um degrau que excede V_A por um valor igual ou maior do que V_T (geralmente 10 a 100 μV). Neste ponto, $\overline{EOC}$ vai para BAIXO e inibe o fluxo de pulsos para o contador, e o contador pára de contar.
7. O processo de conversão agora está completo, conforme sinalizado pela transição de ALTO para BAIXO de $\overline{EOC}$, e o conteúdo do contador é a representação digital de V_A.
8. O contador manterá o valor digital até que o próximo pulso de START inicie uma nova conversão.

EXEMPLO 10-13A

Considere os seguintes valores para o conversor A/D da Fig. 10-13: freqüência do clock = 1 MHz; V_T = 0,1 mV; o conversor D/A tem saída de F.S. = 10,23 V e uma entrada de 10 bits. Determine os seguintes valores.

(a) O equivalente digital obtido para V_A = 3,728 V
(b) O tempo de conversão
(c) A resolução deste conversor

Solução

(a) O conversor D/A tem uma entrada de 10 bits e uma saída de fundo de escala de 10,23 V. Assim, o número total de degraus possíveis é $2^{10} - 1 = 1023$, e portanto o tamanho do degrau é

$$\frac{10{,}23\text{ V}}{1023} = 10\text{ mV}$$

Isto significa que V_{AX} aumenta em degraus de 10 mV conforme o contador avança a partir de 0. Como V_A = 3,728 V e V_T = 0,1 mV, V_{AX} deve alcançar 3,7281 V ou mais para o comparador chavear para BAIXO. Isto demandará

$$\frac{3{,}7281\text{ V}}{10\text{ mV}} = 372{,}81 = 373\text{ degraus}$$

Então, ao final da conversão, o contador manterá o equivalente binário de 373, que é 0101110101. Este é o equivalente digital desejado de V_A = 3,728 V, conforme foi produzido por este conversor A/D.

(b) Trezentos e setenta e três degraus foram necessários para completar a conversão. Logo, 373 pulsos de clock ocorreram numa taxa de um por microssegundo. Isto resulta num tempo total de conversão de 373 μs.

(c) A resolução deste conversor é igual ao tamanho do degrau do conversor D/A, que é 10 mV. Em porcentagem é 1/1023 × 100% ≈ 0,1%.

EXEMPLO 10-13B

Para o mesmo conversor A/D do Exemplo 10-13A, determine a faixa aproximada de tensões analógicas de entrada que produzirá o mesmo resultado digital $0101110101_2 = 373_{10}$.

Solução

A Tabela 10-4 mostra a tensão de saída ideal do conversor D/A, V_{AX}, para vários passos em torno do 373º. Se V_A é li-

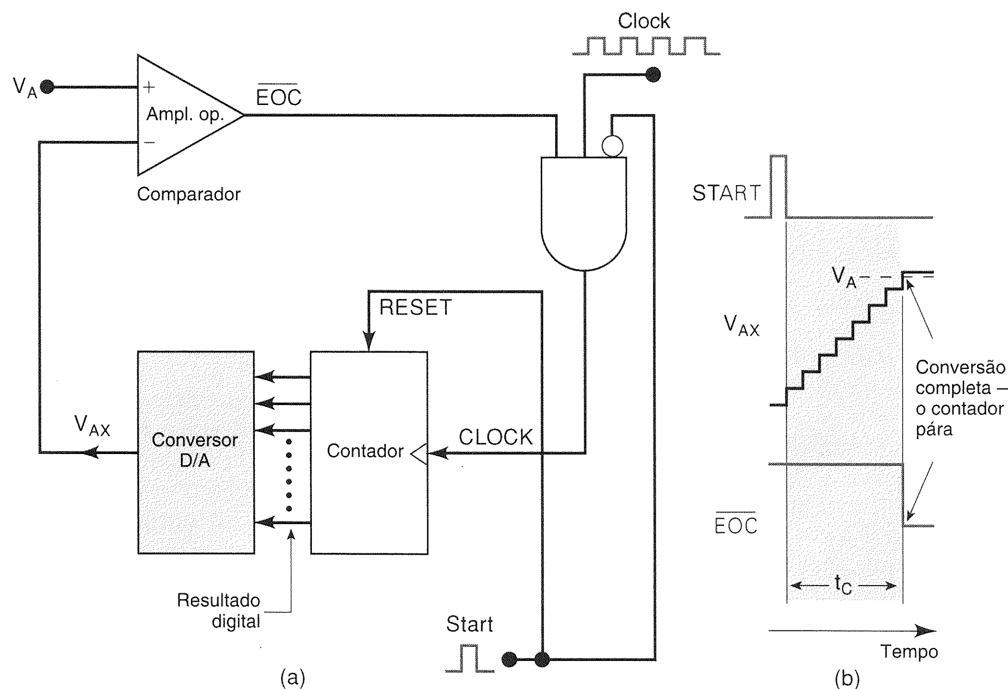


Fig. 10-13 Conversor A/D de rampa digital.

TABELA 10-4

Degrau	V_{AX} (V)
371	3,71
372	3,72
373	3,73
374	3,74
375	3,75

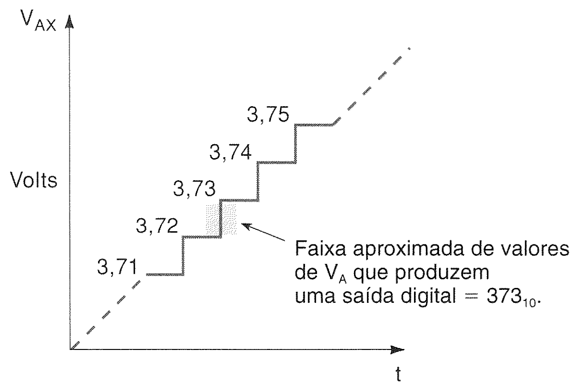


Fig. 10-14 Exemplo 10-13B.

geiramente menor do que 3,72 V (por um valor < V_T), então $\overline{EOC}$ não vai para BAIXO quando V_{AX} alcança o degrau de 3,72 V, mas irá para BAIXO no degrau de 3,73 V. Se V_A é ligeiramente menor do que 3,73 V (por um valor < V_T), então $\overline{EOC}$ não vai para BAIXO até que V_{AX} alcance o degrau de 3,74 V. Assim, enquanto V_A está entre aproximadamente 3,72 V e 3,73 V, $\overline{EOC}$ vai para BAIXO quando V_{AX} alcança o degrau 3,73 V. A faixa exata de valores para V_A é

$$3{,}72\ \text{V} - V_T \quad \text{até} \quad 3{,}73\ \text{V} - V_T$$

mas, como V_T é muito pequeno, podemos simplesmente dizer que a faixa é aproximadamente 3,72 a 3,73 V — uma faixa igual a 10 mV, a resolução do conversor D/A. Isto está ilustrado na Fig. 10-14.

Precisão e Resolução de Conversores A/D

É muito importante compreender os erros associados com a realização de todos os tipos de medições. Uma fonte de erro inevitável no método da rampa digital é que o tamanho do degrau ou resolução do conversor D/A interno é a menor unidade de medida. Imagine tentar medir as alturas de jogadores de basquete, colocando-os em pé próximos a uma escada com degraus de 30 cm e atribuindo aos jogadores a altura do primeiro degrau maior do que a de suas cabeças. Qualquer um com mais do que 1,80 m mediria 2,10 m! Analogamente, a tensão de saída V_{AX} é um forma de onda do tipo escada que aumenta em degraus discretos até exceder a tensão de entrada, V_A. Tornando o tamanho do degrau menor, podemos reduzir o erro potencial, mas sempre haverá uma diferença entre a quantidade real (analógica) e o valor digital associado a ela. Isto é chamado de **erro de quantização**. Assim, V_{AX} é uma aproximação para o valor de V_A, e o melhor que podemos esperar é que V_{AX} esteja no máximo a 10 mV de V_A se a resolução (tamanho do degrau) é 10 mV. Este erro de quantização, que pode ser reduzido aumentando-se o número de bits no contador e no conversor D/A, é por vezes especificado como um erro de +1 LSB, indicando que o resultado pode estar tão diferente quanto o peso do LSB. No Problema 10-28 veremos como este erro de quantização pode ser modificado de modo que seja ±1/2 LSB, que é uma situação mais comum.

Analisando de outro modo, a entrada V_A pode assumir um número *infinito* de valores entre 0 V e o F.S. Entretanto, a aproximação V_{AX} pode assumir apenas um número finito de valores discretos. Isto significa que uma pequena faixa de valores de V_A terá a mesma representação digital. No Exemplo 10-13B, vimos que os valores V_A desde ≈ 3,72 V até ≈ 3,73 V demandarão 373 degraus, o que resulta portanto na mesma representação digital. Em outras palavras, V_A deve mudar de 10 mV (a resolução) para produzir uma alteração na saída digital.

Como no conversor D/A, *precisão* não está relacionada a resolução, mas é dependente da precisão dos componentes do circuito, tais como o comparador, os resistores de precisão e as chaves de corrente dos conversores D/A, as fontes de referência e assim por diante. Uma especificação de erro de 0,01% F.S. indica que o resultado do conversor A/D pode desviar 0,01% F.S., devido a componentes não-ideais. Este erro se *soma* ao erro de quantização devido à resolução. Estas duas fontes de erro usualmente são da mesma ordem de magnitude para um determinado conversor A/D.

EXEMPLO 10-14

Um certo conversor A/D de oito bits tem uma entrada de fundo de escala de 2,55 V (isto é, V_A = 2,55 V produz uma saída digital de 11111111). Ele tem um erro especificado de 0,1% F.S. Determine o valor máximo pelo qual a saída V_{AX} pode diferir da entrada analógica.

Solução

O tamanho do degrau é 2,55 V/(2^8 − 1), que dá exatamente 10 mV. Isto significa que, mesmo que o conversor D/A não apresentasse imprecisões, a saída V_{AX} poderia se desviar até 10 mV porque V_{AX} pode mudar apenas em degraus de 10 mV; este é o erro de quantização. O erro especificado de 0,1% F.S. é 0,1% × 2,55 = 2,55 mV. Isto significa que o valor de V_{AX} pode se desviar até 2,55 mV devido às imprecisões dos componentes. Assim, o erro total possível poderia ser até 10 mV + 2,55 mV = 12,55 mV.

Por exemplo, suponha que a entrada analógica seja 1,268 V. Se a saída do conversor D/A fosse perfeitamente precisa, a escada pararia no degrau 127 (1,27 V). Mas digamos que V_{AX} se desviasse de −2 mV, portanto ele seria 1,268 V no 127º degrau. Isto não seria grande o suficiente para parar a conversão; ela pararia no 128º degrau. Assim, a saída digital seria 10000000_2 = 128_{10} para uma entrada analógica de 1,268 V, um erro de 12 mV.

Tempo de Conversão, t_C

O tempo de conversão é mostrado na Fig. 10-13(b) como o intervalo de tempo entre o fim do pulso de START e a ativação da saída $\overline{EOC}$. O contador inicia a contagem a partir de 0 e conta de modo crescente até que V_{AX} exceda V_A, quando neste ponto $\overline{EOC}$ vai para BAIXO para finalizar o processo de conversão. Deve ficar claro que o valor do tempo de conversão, t_C, depende de V_A. Um valor grande demandará mais degraus antes que a tensão da escada exceda V_A.

O tempo máximo de conversão ocorrerá quando V_A estiver um pouco abaixo do fundo de escala, de modo que V_{AX} terá que ir até o último degrau para ativar $\overline{EOC}$. Para um conversor de N bits, isto seria

$$t_C(\text{max}) = (2^N - 1) \text{ ciclos de clock}$$

Por exemplo, o conversor A/D no Exemplo 10-13A teria um tempo máximo de conversão de

$$t_C(\text{max}) = (2^{10} - 1) \times 1\ \mu\text{s} = 1.023\ \mu\text{s}$$

Algumas vezes, o tempo médio de conversão é especificado; ele é a metade do tempo máximo de conversão. Para um conversor de rampa digital, isto seria

$$t_C(\text{med}) = \frac{t_C(\text{max})}{2} \approx 2^{N-1} \text{ ciclos de clock}$$

A principal desvantagem do método da rampa digital é que o tempo de conversão essencialmente dobra para cada bit que é adicionado ao contador, de modo que a resolução só pode ser melhorada com um aumento de t_C. Isto torna este tipo de conversor A/D inadequado para aplicações em que devem ser feitas conversões A/D repetitivas de sinais analógicos que mudam rapidamente. Entretanto, para aplicações de baixa velocidade, a relativa simplicidade do conversor de rampa digital é uma vantagem sobre os conversores A/D mais complexos e de alta velocidade.

EXEMPLO 10-15

O que acontecerá na operação do conversor A/D de rampa digital se a entrada analógica V_A for maior do que o valor de fundo de escala?

Solução

Consultando a Fig. 10-13, deveria ficar claro que a saída do comparador nunca irá para BAIXO, pois a tensão da escada nunca pode exceder V_A. Assim, pulsos serão continuamente aplicados ao contador, de modo que o contador contará repetidamente de 0 até o máximo, reciclando de volta para 0, contando, e assim por diante. Isto produzirá repetidas formas de onda do tipo escada em V_{AX} indo de 0 até o fundo de escala, e continuará até que V_A seja diminuído para um valor abaixo do fundo de escala.

Questões de Revisão

1. Descreva a operação básica do conversor A/D de rampa digital.
2. Explique o que é *erro de quantização*.
3. Por que o tempo de conversão aumenta com o valor da tensão analógica de entrada?
4. *Verdadeiro ou falso:* Com todo o resto permanecendo igual, um conversor A/D de rampa digital de 10 bits tem uma resolução melhor, mas um tempo de conversão mais longo, do que um conversor A/D de 8 bits.
5. Cite uma vantagem e uma desvantagem de um conversor A/D de rampa digital.
6. Para o conversor do Exemplo 10-13, determine a saída digital para V_A = 1,345 V. Repita para V_A = 1,342 V.

10-10 AQUISIÇÃO DE DADOS

Existem muitas aplicações nas quais os dados analógicos devem ser *digitalizados* (convertidos para digital) e transferidos para a memória de um computador. Este processo pelo qual o computador adquire estes dados analógicos digitalizados é denominado *aquisição de dados*. O computador pode fazer diversas coisas diferentes com os dados, dependendo da aplicação. Numa aplicação de armazenamento, como gravação digital de áudio, gravação de vídeo ou um osciloscópio digital, o microcomputador interno armazenará os dados e então poderia transferi-los mais tarde para um conversor D/A para reproduzir o sinal analógico original. Numa aplicação de controle de processos, o computador pode examinar os dados e realizar cálculos com os mesmos para determinar as saídas de controle que deve gerar.

A Fig. 10-15(a) mostra como um microcomputador é conectado em um conversor A/D de rampa digital para aquisição de dados. O computador gera os pulsos de START que iniciam cada nova conversão A/D. O sinal $\overline{EOC}$ (fim de conversão) do conversor A/D é levado para o computador. O computador monitora $\overline{EOC}$ para saber quando a conversão A/D está completa; então transfere o dado digital da saída do conversor A/D para sua memória.

As formas de onda na Fig. 10-15(b) ilustram como o computador adquire uma versão digital do sinal analógico, V_A. A forma de onda do tipo escada, V_{AX}, que é gerada internamente ao conversor A/D, é apresentada sobreposta à forma de onda de V_A para fins de ilustração. O processo inicia em t_0, quando o computador gera um pulso de START para iniciar um ciclo de conversão A/D. A conversão é concluída em t_1, quando a escada excede V_A, e $\overline{EOC}$ vai para BAIXO. Esta descida de $\overline{EOC}$ sinaliza ao computador que o conversor A/D tem uma saída digital que representa o valor de V_A no ponto *a*, e o computador guardará este dado na memória.

O computador gera um novo pulso de START logo após t_1 para iniciar o segundo ciclo de conversão. Note que isto resseta a escada para 0 e $\overline{EOC}$ retorna para ALTO porque o pulso de START resseta o contador do conversor A/D. O segundo ciclo de conversão termina em t_2, quando a escada novamente excede V_A. O computador então armazena na memória o dado digital correspondente ao ponto *b*. Estas etapas são repetidas em t_3, t_4 e assim por diante.

O processo no qual o computador gera um pulso de START, monitora $\overline{EOC}$ e armazena o dado do conversor A/

D na memória é feito sob o controle de um programa que o computador está executando. Este programa de aquisição de dados determinará quantos dados obtidos do sinal analógico serão armazenados na memória do computador.

Reconstruindo um Sinal Digitalizado

Na Fig. 10-15(b), o conversor A/D está operando na sua máxima velocidade, pois um novo pulso de START é gerado imediatamente após o computador adquirir o dado de saída do conversor A/D da conversão anterior. Note que os tempos de conversão não são constantes, porque o valor da entrada analógica está mudando. O problema com este método de armazenar uma forma de onda é que, para reconstruí-la, necessitaríamos saber os instantes de tempo nos quais cada valor deveria ser reproduzido. Normalmente, quando se armazena uma forma de onda digitalizada, as amostras são obtidas em intervalos fixos a uma taxa pelo menos duas vezes maior do que a freqüência mais alta contida no sinal analógico. O sistema digital armazena a forma de onda como uma lista de valores de dados amostrados. A Tabela 10-5 mostra uma lista de dados que poderia estar armazenada se o sinal na Fig. 10-16(a) fosse digitalizado.

Na Fig. 10-16(a) vemos como o conversor A/D realiza conversões continuamente para digitalizar o sinal analógico

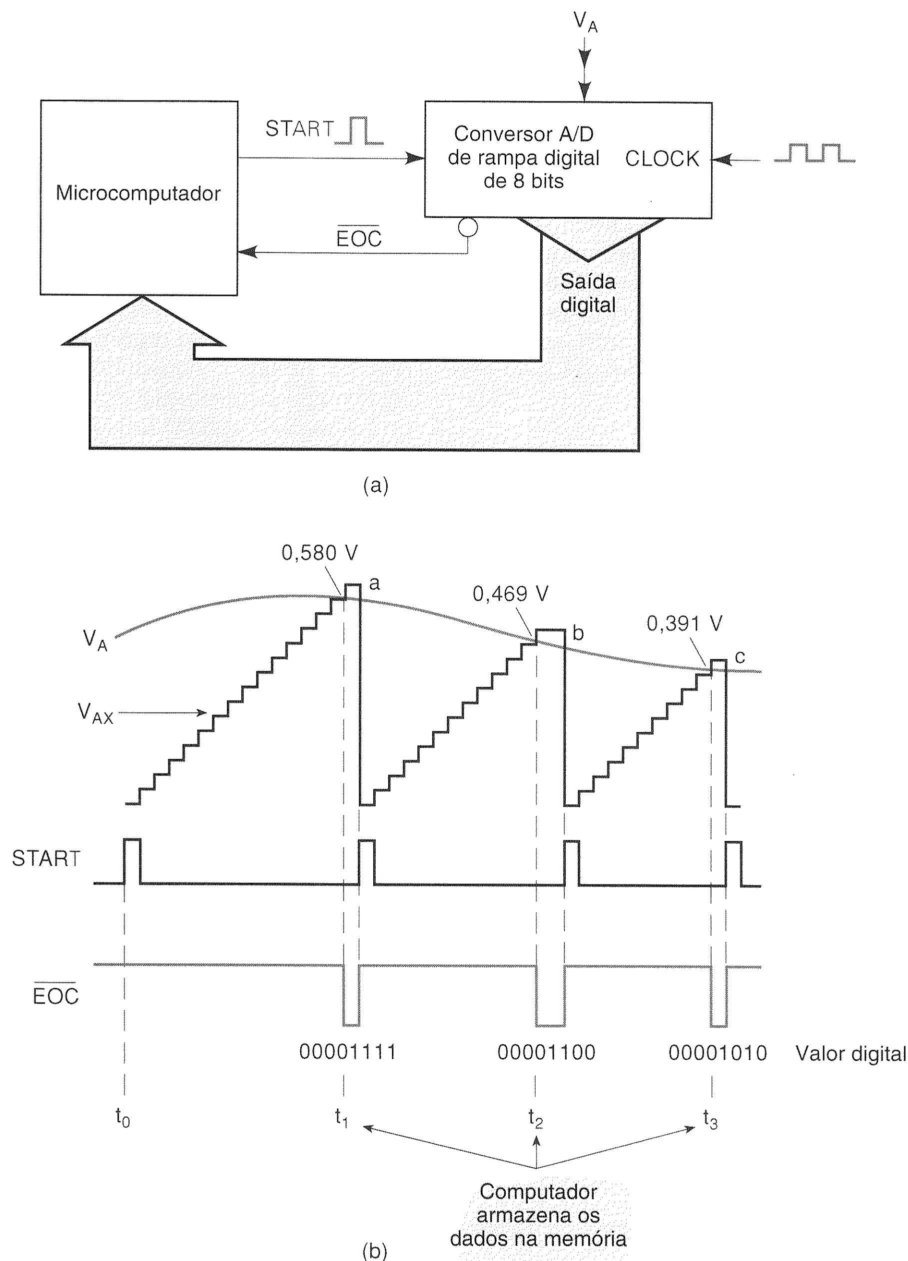


Fig. 10-15 (a) Típico sistema computacional de aquisição de dados; (b) formas de onda mostrando como o computador inicia cada novo ciclo de conversão e no final armazena o dado digital na memória.

TABELA 10-5 Amostras de dados digitalizados

Ponto	Tensão Real (V)	Equivalente Digital
a	1,22	01111010
b	1,47	10010011
c	1,74	10101110
d	1,70	10101010
e	1,35	10000111
f	1,12	01110000
g	0,91	01011011
h	0,82	01010010

nos pontos *a*, *b*, *c*, *d* e assim por diante. Se estes dados digitais forem usados para reconstruir o sinal, o resultado será parecido com o da Fig. 10-16(b). A linha do tipo escada representa a forma de onda da tensão que sairia do conversor D/A. A outra linha seria o resultado obtido ao passar o sinal por um filtro passa-baixa RC simples. Podemos constatar que esta é uma reprodução bastante boa do sinal analógico original. Isto é porque o sinal analógico não faz nenhuma alteração rápida entre os pontos digitalizados. Se o sinal analógico contivesse variações de alta freqüência, o conversor A/D não seria capaz de acompanhar as variações, e a versão reproduzida seria muito menos precisa. Por esta razão, é importante manter o tempo de conversão do conversor A/D pequeno, de modo que o sinal analógico não se altere significativamente entre as conversões. Isto enfatiza a necessidade de conversores A/D com tempos de conversão menores do que os dos conversores simples de rampa digital. Examinaremos métodos de conversão A/D muito mais rápidos nas próximas seções.

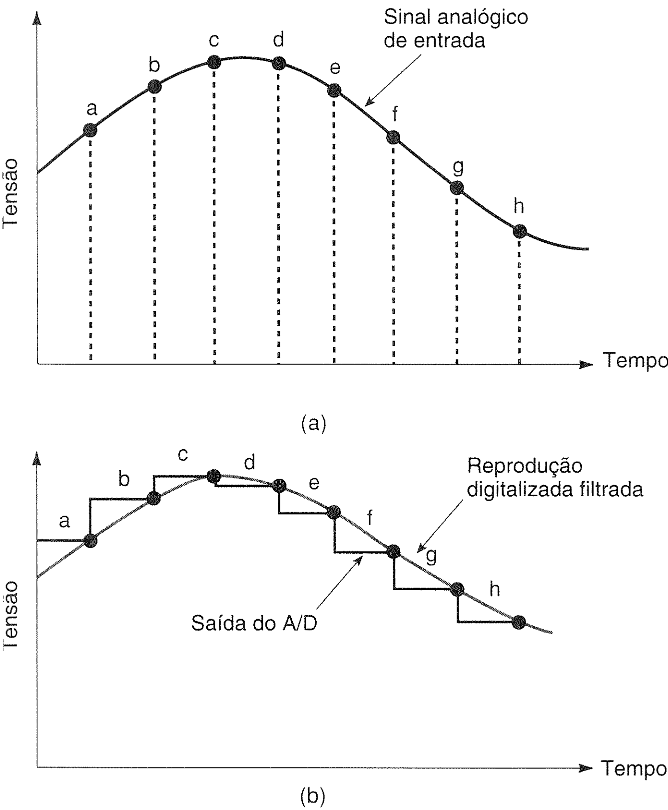


Fig. 10-16 (a) Digitalização de um sinal analógico; (b) reconstruindo o sinal a partir dos dados digitais.

Questões de Revisão

1. O que é *digitalizar um sinal*?
2. Descreva as etapas em um processo computacional de aquisição de dados.

10-11 CONVERSOR A/D DE APROXIMAÇÕES SUCESSIVAS

O **conversor de aproximações sucessivas** é um dos tipos de conversor A/D mais utilizados. Ele tem circuitos mais complexos do que o conversor A/D de rampa digital mas tempo de conversão muito menor. Além disso, conversores de aproximações sucessivas têm um tempo de conversão fixo que não é dependente do valor da entrada analógica.

A configuração básica, mostrada na Fig. 10-17(a), é similar à do conversor A/D de rampa digital. Entretanto, o conversor A/D de aproximações sucessivas não utiliza um contador para fornecer a entrada do bloco conversor D/A; em vez disso, ele usa um registrador. A lógica de controle modifica o conteúdo do registrador bit a bit até que o dado do registrador seja o equivalente digital da entrada analógica V_A, dentro da resolução do conversor. A seqüência básica de operação é apresentada no fluxograma na Fig. 10-17(b). Vamos seguir este fluxograma à medida que analisamos o exemplo ilustrado na Fig. 10-18.

Para este exemplo, escolhemos um conversor simples de quatro bits com tamanho de degrau de 1 V. Embora a maioria deste tipo de conversor na prática tenha mais bits e melhor resolução que nosso exemplo, a operação será exatamente a mesma. Neste ponto, você já deveria ser capaz de determinar que os quatro bits do registrador que acionam o conversor D/A têm pesos de 8, 4, 2 e 1 V, respectivamente.

Vamos considerar que a entrada analógica é V_A = 10,4 V. A operação inicia com a lógica de controle limpando todos os bits do registrador para 0, de modo que $Q_3 = Q_2 = Q_1 = Q_0 = 0$. Expressaremos isto como [Q] = 0000. Isto faz a saída do conversor D/A $V_{AX} = 0$, conforme indicado no instante t_0 do diagrama de tempo na Fig. 10-18. Com $V_{AX} < V_A$, a saída do comparador está em ALTO.

No próximo passo (instante t_1), a lógica de controle ajusta o MSB do registrador para 1, de modo que [Q] = 1000. Isto produz V_{AX} = 8 V. Como $V_{AX} < V_A$, a saída COMP ainda está em ALTO. Este nível ALTO informa a lógica de controle que o ajuste do MSB não fez V_{AX} exceder V_A, e portanto o MSB é mantido em 1.

Agora a lógica de controle atua sobre o próximo bit menos significativo, Q_2. Ela leva Q_2 a 1 para produzir [Q] = 1100 e V_{AX} = 12 V no instante t_2. Como $V_{AX} > V_A$, a saída COMP vai para BAIXO. Este nível BAIXO sinaliza para a lógica de controle que o valor de V_{AX} está muito grande, e então a lógica de controle limpa Q_2 de volta para 0 em t_3. Assim, em t_3, o conteúdo do registrador volta para 1000 e V_{AX} retorna para 8 V.

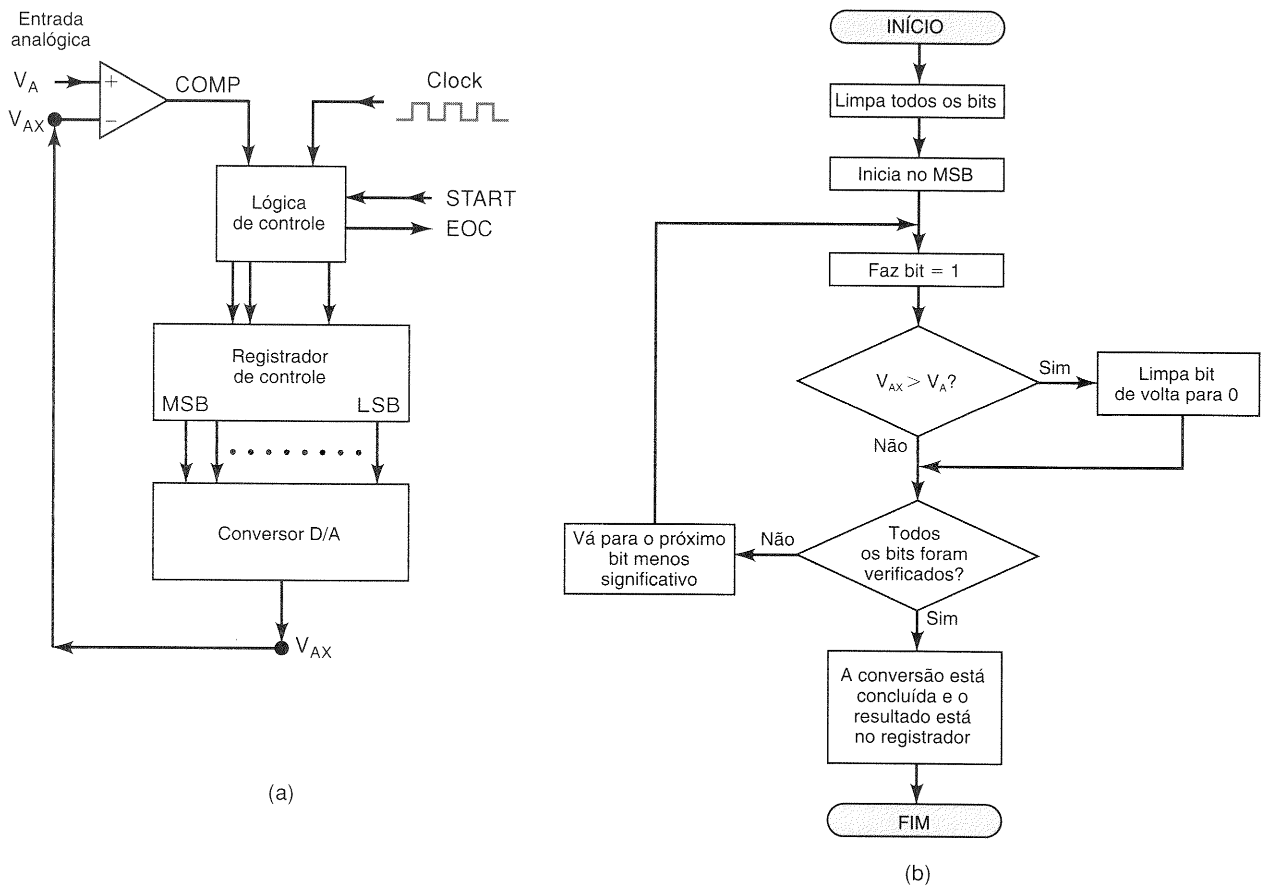


Fig. 10-17 Conversor A/D de aproximações sucessivas; (a) diagrama de blocos simplificado; (b) fluxograma de operação.

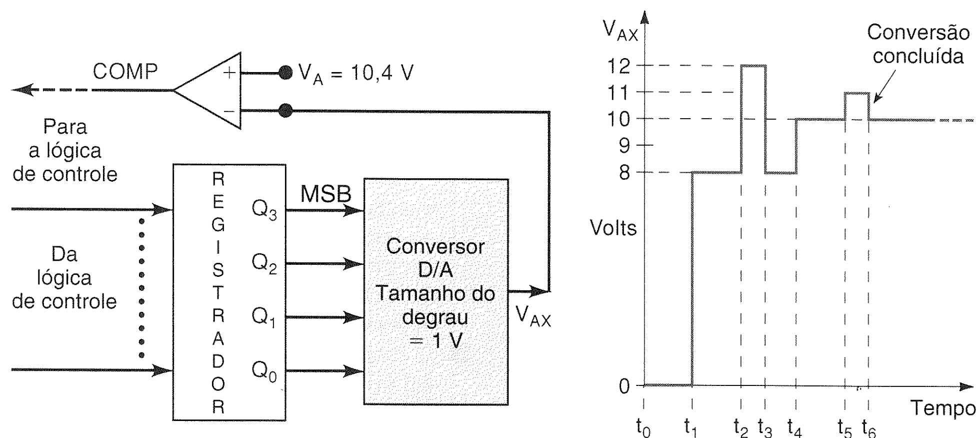


Fig. 10-18 Ilustração da operação de um conversor A/D de aproximações sucessivas de quatro bits utilizando um conversor D/A com tamanho de degrau de 1 V e V_A = 10,4 V.

A próxima etapa ocorre em t_4, onde a lógica de controle ajusta o próximo bit menos significativo, Q_1, para 1, de modo que $[Q] = 1010$ e $V_{AX} = 10$ V. Com $V_{AX} < V_A$, COMP fica em ALTO e faz a lógica de controle manter Q_1 em 1.

O passo final ocorre em t_5, onde a lógica de controle ajusta o próximo bit menos significativo, Q_0, para 1, de modo que $[Q] = 1011$ e $V_{AX} = 11$ V. Como $V_{AX} > V_A$, COMP vai para ALTO para sinalizar que V_{AX} é muito grande, e a lógica de controle limpa Q_0 de volta para 0 em t_6.

Neste ponto, todos os bits do registrador foram processados, a conversão está concluída, e a lógica de controle ativa a saída $\overline{EOC}$ para sinalizar que o equivalente digital de V_A está no registrador. Para este exemplo, a saída digital para $V_A = 10{,}4$ V é $[Q] = 1010$. Note que 1010 equivale na verdade a 10 V, que é *menor do que* a entrada analógica; esta é uma característica do método de aproximações sucessivas. Lembre-se de que no método da rampa digital a saída digital era sempre equivalente a uma tensão que estava no degrau acima de V_A.

EXEMPLO 10-16

Um conversor A/D de aproximações sucessivas de oito bits tem uma resolução de 20 mV. Qual será a sua saída digital para uma entrada analógica de 2,17 V?

Solução

$$2{,}17 \text{ V}/20 \text{ mV} = 108{,}5$$

de modo que o degrau 108 produziria $V_{AX} = 2{,}16$ V e o degrau 109 produziria 2,18 V. O conversor sempre produz um V_{AX} final que está um degrau *abaixo* de V_A. Portanto, para o caso de $V_A = 2{,}17$ V, o resultado digital seria $108_{10} = 01101100_2$.

Tempo de Conversão

Na operação recém-descrita, a lógica de controle atua sobre cada bit do registrador, ajusta-o para 1, decide se deve ou não mantê-lo em 1 e passa para o próximo bit. O processamento de cada bit dura um ciclo de clock, de modo que o tempo de conversão total para um conversor A/D de aproximações sucessivas de N bits será de N ciclos de clock. Isto é,

$$\begin{matrix} t_C \text{ para o conversor A / D} \\ \text{de aproximações sucessivas} \end{matrix} = N \times 1 \text{ ciclos de clock}$$

Este tempo de conversão será o mesmo, *não importando o valor de V_A*. Isto acontece porque a lógica de controle processa cada bit para verificar se o 1 é ou não necessário.

EXEMPLO 10-17

Compare os tempos máximos de conversão de um conversor A/D de rampa digital de 10 bits com um conversor A/D de aproximações sucessivas de 10 bits, no caso de ambos utilizarem a freqüência de clock de 500 kHz.

Solução

Para o conversor de rampa digital, o tempo máximo de conversão é

$$(2^N - 1) \times (1 \text{ ciclo de clock}) = 1.023 \times 2\ \mu\text{s} = 2.046\ \mu\text{s}$$

Para um conversor de aproximações sucessivas de 10 bits, o tempo de conversão sempre é 10 períodos de clock ou

$$10 \times 2\ \mu\text{s} = 20\ \mu\text{s}$$

Logo, ele é aproximadamente 100 vezes mais rápido do que o conversor de rampa digital.

Tendo em vista que os conversores A/D de aproximações sucessivas têm tempos de conversão relativamente rápidos, sua utilização em aplicações de aquisição de dados permite que mais dados sejam adquiridos em um certo intervalo de tempo. Isto pode ser muito importante quando o dado analógico está mudando a uma taxa relativamente alta.

Como diversos conversores desse tipo estão disponíveis como CIs, raramente é necessário projetar o circuito da lógica de controle, e portanto não abordaremos isso. Para aqueles que estiverem interessados em detalhes sobre a lógica de controle, os manuais dos fabricantes devem apresentar detalhes suficientes.

Um CI Real: O Conversor A/D de Aproximações Sucessivas ADC0804

Conversores A/D estão disponíveis de vários fabricantes de CIs com uma ampla faixa de características de operação. Analisaremos um dos dispositivos mais populares para se ter uma idéia do que é realmente usado nas aplicações em sistemas. A Fig. 10-19 é a configuração de pinos para o ADC0804, que é um CI de 20 pinos que realiza conversão A/D utilizando o método de aproximações sucessivas. Algumas de suas características mais importantes são as seguintes:

- Tem duas entradas analógicas, $V_{IN}(+)$ e $V_{IN}(-)$, para permitir **entradas diferenciais**. Em outras palavras, a entrada analógica real, V_{IN}, é a diferença das tensões aplicadas nestes pinos [V_{IN} analógico $= V_{IN}(+) - V_{IN}(-)$]. Para medições comuns, não-diferenciais, a entrada analógica é aplicada em $V_{IN}(+)$, enquanto $V_{IN}(-)$ é conectado no terra analógico. Durante sua operação normal, o conversor utiliza $V_{CC} = +5$ V como sua tensão de referência, e a entrada analógica pode variar de 0 até 5 V de fundo de escala.
- Converte a tensão analógica de entrada em uma saída digital de oito bits. A saída digital tem um buffer tristate, de modo que pode ser facilmente conectada numa barra de dados. Com oito bits, a resolução é 5 V / 255 = 19,6 mV.
- Possui um circuito gerador de clock interno que produz uma freqüência de $f = 1/(1{,}1RC)$, onde R e C são valores de componentes externos. Uma freqüência de clock típica é 606 kHz utilizando $R = 10$ kΩ e $C = 150$ pF. Também pode ser usado um sinal externo de clock, conectando-o no pino CLK IN.

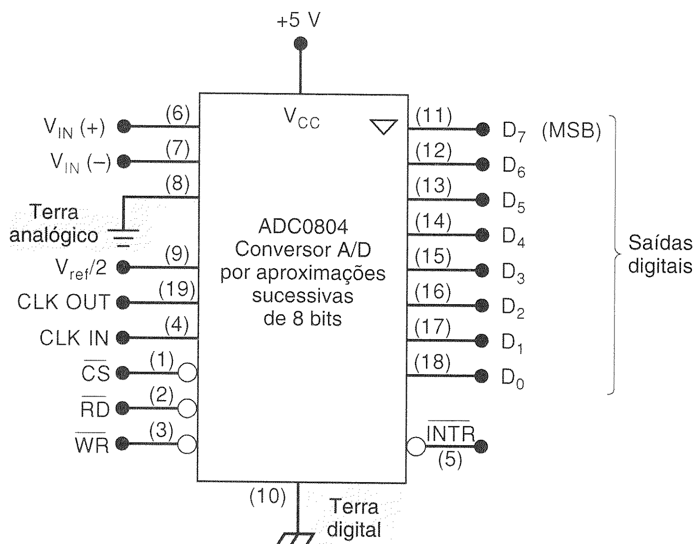


Fig. 10-19 Conversor A/D de aproximações sucessivas com saídas tristate ADC0804. Os números entre parênteses são os números dos pinos do CI.

- Utilizando uma freqüência de clock de 606 kHz, o tempo de conversão é de aproximadamente 100 μs.
- Ele possui conexões de terra separadas para tensões digitais e analógicas. O pino 8 é o terra analógico que é conectado ao ponto de referência comum do circuito analógico que está gerando a tensão analógica. O pino 10 é o terra digital que é o utilizado por todos os dispositivos digitais do sistema. (Note os símbolos diferentes utilizados para os terras diferentes.) O terra digital é inerentemente ruidoso devido às rápidas mudanças de corrente que ocorrem quando os dispositivos digitais mudam de estado. Embora não seja necessário usar um terra analógico separado, fazer desta maneira garante que o ruído do terra digital é impedido de causar o chaveamento prematuro do comparador analógico dentro do conversor A/D.

Este CI foi projetado para ser facilmente interfaceado com a barra de dados de um microprocessador. Por isso, os nomes de algumas entradas e saídas do ADC0804 são baseados nas funções que são comuns nos sistemas microprocessados. As funções destas entradas e saídas são definidas a seguir:

- $\overline{CS}$ **(Chip Select** — Seleção do Chip). Esta entrada deve estar no estado ativo em BAIXO para que as entradas $\overline{RD}$ ou $\overline{WR}$ tenham efeito. Com $\overline{CS}$ em ALTO, as saídas digitais estão no estado de alta impedância, e nenhuma conversão pode ser realizada.
- $\overline{RD}$ **(READ** — LEITURA). Este sinal é utilizado para habilitar os buffers da saída digital. Com $\overline{CS}$ = $\overline{RD}$ = BAIXO, os pinos da saída digital terão os níveis lógicos representativos do resultado da *última* conversão A/D. O microcomputador pode então *ler* este valor digital pelo barramento de dados.
- $\overline{WR}$ **(WRITE** — ESCRITA). Um pulso em BAIXO é aplicado nesta entrada para iniciar uma nova conversão. É na verdade uma entrada de início de conversão. É chamada de entrada de **escrita** porque, numa aplicação típica, o microcomputador gera um pulso de escrita (similar ao usado para escrever na memória) que aciona esta entrada.
- $\overline{INTR}$ **(INTERRUPT** — INTERRUPÇÃO). Este sinal de saída vai para ALTO no início da conversão e retornará para BAIXO para sinalizar o fim da conversão. Na verdade, é um sinal de saída de fim de conversão, mas é chamado de interrupção pois numa situação típica ele é enviado para a entrada de interrupção do microprocessador para obter a sua atenção e informá-lo de que o dado do conversor está pronto para ser lido.
- $V_{ref}/2$**.** Esta é uma entrada opcional que pode ser usada para reduzir a tensão de referência interna e portanto para mudar a faixa analógica de entrada que o conversor pode tratar. Quando esta entrada está desconectada, ela assume 2,5 V ($V_{CC}/2$), já que V_{CC} está sendo usado como tensão de referência. Conectando-se uma tensão externa neste pino, a referência interna é alterada para o dobro desta tensão, e a faixa analógica de entrada se altera de modo correspondente. A Tabela 10-6 ilustra isto.

TABELA 10-6

$V_{ref}/2$	Faixa Analógica de Entrada (V)	Resolução (mV)
Em aberto	0-5	19,6
2,25	0-4,5	17,6
2,0	0-4	15,7
1,5	0-3	11,8

- **CLK OUT.** Um resistor é conectado neste pino para utilização do clock interno. O sinal de clock aparece neste pino.
- **CLK IN.** Usado como entrada do clock externo, ou para a conexão de um capacitor quando se utiliza clock interno.

A Fig. 10-20(a) mostra uma conexão típica do ADC0804 com um microprocessador numa aplicação de aquisição de dados. O microprocessador controla quando a conversão é realizada gerando os sinais de $\overline{CS}$ e $\overline{WR}$. Então ele adquire os dados de saída do conversor A/D, gerando os sinais $\overline{CS}$ e $\overline{RD}$ após detectar uma transição de descida em $\overline{INTR}$, indicando o fim da conversão. As formas de onda na Fig. 10-20(b) mostram a atividade dos sinais durante o processo de aquisição de dados. Note que $\overline{INTR}$ vai para ALTO quando $\overline{CS}$ e $\overline{WR}$ estão em BAIXO, mas o processo de conversão não se inicia até que $\overline{WR}$ retorne para ALTO. Note também que as linhas de saída do conversor A/D ficam no estado de alta impedância até que o microprocessador ative $\overline{CS}$ e $\overline{RD}$; neste ponto, os buffers de dados do conversor A/D são habilitados, de modo que os dados do conversor A/D são enviados para o microprocessador pelo barramento de dados. As linhas de dados retornam para o estado de alta impedância quando ou $\overline{CS}$ ou $\overline{RD}$ retornar para ALTO.

Nesta aplicação do ADC0804, o sinal de entrada está variando numa faixa de 0,5 até 3,5 V. De modo a aproveitar bem os oito bits de resolução, o A/D deve ser compatibili-

zado com a especificação do sinal de entrada. Neste caso, a faixa de fundo de escala é 3,0 V. Entretanto, ela é deslocada da terra por 0,5 V. O deslocamento de 0,5 V é aplicado na entrada negativa $V_{IN}(-)$, estabelecendo-o como o valor 0 de referência. A faixa de 3,0 V é ajustada aplicando-se 1,5 V em $V_{ref}/2$, o que estabelece V_{ref} como 3,0 V. Uma entrada de 0,5 V produzirá um valor digital 00000000, e uma entrada de 3,5 V produzirá 11111111.

Um outro cuidado importante quando interfaceamos sinais analógicos e digitais é o *ruído*. Repare que os caminhos dos terras digital e analógico estão separados. Os dois terras são conectados juntos num ponto que está muito próximo do conversor A/D. Um caminho de resistência extremamente baixa liga este ponto diretamente ao terminal da fonte de alimentação. Também é aconselhável rotear separadamente linhas positivas da fonte de alimentação para os dispositivos digitais e analógicos e utilizar capacitores de desacoplamento (0,01 μF) bem próximos das conexões de alimentação de cada chip.

Questões de Revisão

1. Qual é a principal vantagem de um conversor A/D por aproximações sucessivas sobre um conversor A/D de rampa digital?

Fig. 10-20 (a) Uma aplicação do ADC0804; (b) sinais típicos de temporização durante a aquisição de dados.

2. Qual é a principal desvantagem de um conversor A/D por aproximações sucessivas sobre um conversor A/D de rampa digital?
3. *Verdadeiro ou falso:* O tempo de conversão de um conversor A/D por aproximações sucessivas aumenta conforme a tensão analógica de entrada aumenta.
4. Responda as seguintes questões relativas ao ADC0804.
   (a) Qual é a sua resolução em bits?
   (b) Qual é a faixa normal de tensão analógica de entrada?
   (c) Descreva as funções das entradas $\overline{CS}$, $\overline{WR}$ e $\overline{RD}$.
   (d) Qual é a função da saída $\overline{INTR}$?
   (e) Por que ele tem dois terras separados?
   (f) Qual é o objetivo de $V_{IN}(-)$?

10-12 CONVERSOR A/D FLASH

O **conversor flash** é o conversor A/D disponível de maior velocidade, mas requer muito mais circuitos que os outros tipos. Por exemplo, um conversor A/D flash de seis bits necessita de 63 comparadores analógicos, enquanto uma unidade de oito bits requer 255 comparadores e um conversor de dez bits necessita de 1023 comparadores. O grande número de comparadores limitou o tamanho dos conversores do tipo flash. Conversores flash em CIs comumente são disponíveis em unidades de dois a oito bits, e a maioria dos fabricantes oferece também unidades com nove e dez bits.

O princípio de operação será descrito para um conversor flash de três bits de modo a manter o circuito de um tamanho razoável. Uma vez que o conversor de três bits tenha sido entendido, será fácil estender a idéia básica para conversores flash com mais bits.

O conversor flash na Fig. 10-21(a) tem três bits de resolução e 1 V para tamanho do degrau. O divisor de tensão fornece os níveis de referência de cada comparador, de modo que existem sete níveis correspondentes a 1 V (peso do LSB), 2 V, 3 V, ... e 7 V (fundo de escala). A entrada analógica, V_A, está conectada na outra entrada de cada comparador.

Com $V_A < 1$ V, todas as saídas dos comparadores, de C_1 até C_7, estarão em ALTO. Com $V_A > 1$ V, uma ou mais saídas dos comparadores estarão em BAIXO. As saídas dos comparadores são conectadas num codificador de prioridade de entradas ativas em BAIXO que gera uma saída binária correspondente à saída do comparador de mais alta ordem que estiver em BAIXO. Por exemplo, quando V_A está entre 3 e 4 V, as saídas C_1, C_2 e C_3 estarão em BAIXO e todas as outras em ALTO. O codificador de prioridade responderá apenas ao nível BAIXO da entrada C_3 e produzirá uma saída binária $CBA = 011$, que representa o equivalente digital de V_A, dentro da resolução de 1 V. Quando V_A é maior do que 7 V, C_1 até C_7 estarão todos em BAIXO, e o codificador vai produzir $CBA = 111$ como sendo o equivalente digital da entrada analógica. A Tabela na Fig. 10-21(b) mostra as respostas para todos os valores possíveis de entrada analógica.

O conversor A/D flash da Fig. 10-21 tem uma resolução de 1 V porque a entrada analógica deve mudar 1 V para levar a saída digital para seu próximo valor. Para atingir melhores resoluções, teríamos que aumentar o número de níveis de tensão (isto é, utilizar mais resistores no divisor de tensão) e o número de comparadores. Por exemplo, um conversor flash de oito bits requisitaria $2^8 = 256$ níveis de tensão, incluindo 0 V. Isto demandaria 256 resistores e 255 comparadores (não existe comparador para o nível de 0 V). As 255 saídas dos comparadores acionariam um circuito codificador de prioridade que produziria um código de oito bits correspondente à saída do comparador de mais alta ordem que estivesse em BAIXO. De um modo geral, um conversor flash de N bits requer $2^N - 1$ comparadores, 2^N resistores e a lógica codificadora necessária.

Tempo de Conversão

O conversor flash não utiliza sinal de clock porque não é necessária nenhuma temporização ou seqüenciamento. A conversão se realiza continuamente. Quando o valor da entrada analógica muda, as saídas dos comparadores mudarão, causando assim a mudança das saídas do codificador. O tempo de conversão é o tempo necessário para o aparecimento de uma nova saída digital em resposta a uma mudança em V_A, e depende apenas dos atrasos de propagação dos comparadores e da lógica do codificador. Por isso, os conversores flash possuem tempos de conversão extremamente reduzidos. Por exemplo, o MC10319 da Motorola é um conversor A/D que utiliza internamente circuitos ECL de alta velocidade mas tem entradas e saídas lógicas compatíveis com TTL. Seu tempo de conversão típico é menor do que 20 ns. O AD9010 da Analog Devices é um conversor flash de 10 bits com um tempo de conversão inferior a 15 ns.

Questões de Revisão

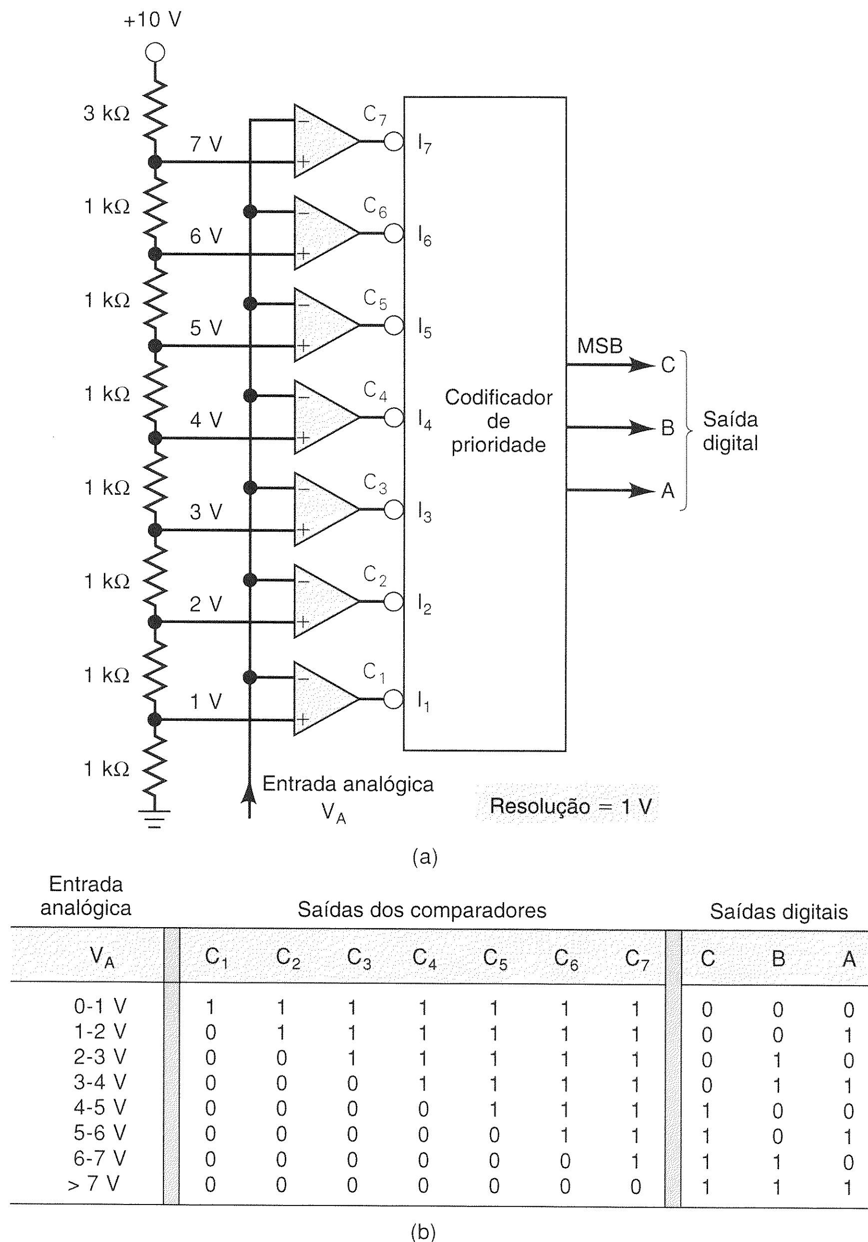
1. *Verdadeiro ou falso:* Um conversor A/D flash não contém um conversor D/A.
2. Quantos comparadores seriam necessários para um conversor flash de 12 bits? Quantos resistores?
3. Cite a principal vantagem e a principal desvantagem de um conversor flash.

10-13 OUTROS MÉTODOS DE CONVERSÃO A/D

Muitos outros métodos de conversão A/D têm sido utilizados há algum tempo, cada qual com suas vantagens e desvantagens. Descreveremos resumidamente alguns deles a seguir.

Conversor A/D de Rampa Digital Crescente/Decrescente (Conversor A/D Rastreador)

Como estudamos anteriormente, o conversor A/D de rampa digital é relativamente lento porque o contador é ressetado para 0 no início de cada nova conversão. A escada sempre começa em 0 V e avança degrau a degrau até o "ponto de chaveamento", onde V_{AX} excede V_A e a saída do comparador comuta para BAIXO. O tempo necessário para



Entrada analógica V_A	Saídas dos comparadores C_1	C_2	C_3	C_4	C_5	C_6	C_7	Saídas digitais C	B	A
0-1 V	1	1	1	1	1	1	1	0	0	0
1-2 V	0	1	1	1	1	1	1	0	0	1
2-3 V	0	0	1	1	1	1	1	0	1	0
3-4 V	0	0	0	1	1	1	1	0	1	1
4-5 V	0	0	0	0	1	1	1	1	0	0
5-6 V	0	0	0	0	0	1	1	1	0	1
6-7 V	0	0	0	0	0	0	1	1	1	0
> 7 V	0	0	0	0	0	0	0	1	1	1

(b)

Fig. 10-21 (a) Conversor A/D flash de três bits; (b) tabela-verdade.

ressetar a escada para 0 e retornar novamente para o novo ponto de chaveamento é realmente desperdiçado. O **conversor A/D de rampa digital crescente/decrescente** utiliza um contador crescente/decrescente para reduzir este tempo desperdiçado.

O contador crescente/decrescente substitui o contador que aciona o conversor D/A. Ele é projetado para contar no modo crescente sempre que a saída do comparador indicar que $V_{AX} < V_A$, e para contagem decrescente quando $V_{AX} > V_A$. Assim, a saída do D/A está sempre sendo dirigida na direção do valor de V_A. Cada vez que a saída do comparador comuta de estado, ela indica que V_{AX} "cruzou" o valor de V_A, o equivalente digital de V_A está no contador e a conversão está completa.

Quando uma nova conversão deve começar, o contador *não é ressetado para 0*, apenas começa a contagem crescente ou decrescente a partir do seu último valor, dependendo da saída do comparador. Ele vai contar até que a escada atravesse V_A de novo para terminar a conversão. Então, a forma de onda de V_{AX} conterá degraus para cima e para baixo que "rastrearão" o sinal V_A. Por isso, este conversor é freqüentemente chamado de *conversor A/D rastreador*.

É evidente que os tempos de conversão geralmente serão reduzidos porque o contador não começa do 0 toda vez mas simplesmente conta crescente ou decrescente a partir do valor anterior. Naturalmente, o valor de t_C ainda dependerá do valor de V_A, e portanto não será constante.

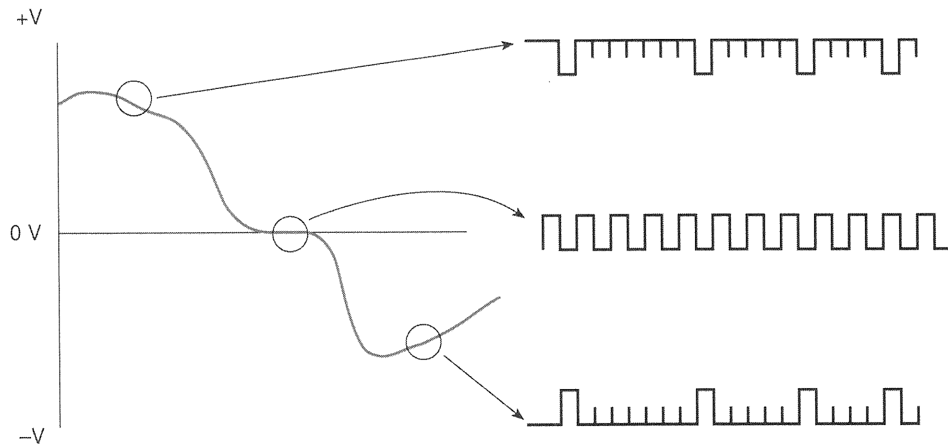


Fig. 10-22 Modulação sigma/delta.

Conversor A/D de Rampa Dupla

O **conversor de rampa dupla** tem um dos tempos de conversão mais longos (geralmente de 10 a 100 ms), mas tem a vantagem do custo relativamente baixo porque não requer componentes de precisão tal como um conversor D/A ou um VCO. A operação básica deste conversor envolve a carga e descarga *linear* de um capacitor utilizando correntes constantes. Primeiro, o capacitor é carregado num intervalo de tempo fixo por uma corrente constante derivada da tensão analógica de entrada, V_A. Assim, ao final deste intervalo de carga fixo, a tensão do capacitor será proporcional a V_A. Neste ponto, o capacitor é descarregado linearmente por uma corrente constante derivada de uma tensão de referência precisa, V_{ref}. Quando a tensão do capacitor alcança 0, a descarga linear termina. Durante o intervalo de descarga, uma freqüência digital de referência é levada para um contador e contada. A duração do intervalo de descarga será proporcional à tensão inicial do capacitor. Assim, no final do intervalo de descarga, o contador terá um valor proporcional à tensão inicial do capacitor, que, como dissemos, é proporcional à V_A.

Além do seu baixo custo, uma outra vantagem do conversor A/D de rampa dupla é a sua baixa sensibilidade ao ruído e às variações em seus componentes devidas a mudanças de temperatura. Por causa do seu longo tempo de conversão, o conversor A/D de rampa dupla não é utilizado em aplicações de aquisição de dados. Entretanto, o tempo de conversão grande não é um problema em aplicações como multímetros ou voltímetros digitais, e é onde eles encontram sua principal aplicação.

Conversor A/D de Tensão-Freqüência

O **conversor A/D de tensão-freqüência** é mais simples do que outros conversores A/D porque não utiliza um conversor D/A. Em vez disso, ele usa um *oscilador controlado à tensão linear (VCO — voltage-controlled oscillator)* que produz uma freqüência de saída que é proporcional à sua tensão de entrada. A tensão analógica que deve ser convertida é aplicada ao VCO para gerar uma freqüência de saída. Esta freqüência é levada para um contador para ser contada por um intervalo de tempo fixo. A contagem final é proporcional ao valor da tensão analógica.

Para ilustrar, suponha que o VCO gera uma freqüência de 10 kHz para cada volt de entrada (isto é, 1 V produz 10 kHz, 1,5 V produz 15 kHz, 2,73 V produzem 27,3 kHz). Se a tensão analógica de entrada é 4,54 V, então a saída do VCO será um sinal de 45,4 kHz que aciona um contador por, digamos, 10 ms. Após um intervalo de contagem de 10 ms, o contador terá uma contagem de 454, que é a representação digital de 4,54 V.

Embora este seja um método de conversão simples, é difícil atingir um alto grau de precisão devido à dificuldade em projetar VCOs com precisão melhor do que 0,1 por cento.

Uma das principais aplicações desse tipo de conversor está em ambientes industriais ruidosos onde pequenos sinais analógicos devem ser transmitidos de circuitos transdutores para um computador de controle. Os pequenos sinais analógicos podem ser drasticamente afetados pelo ruído se forem transmitidos diretamente para o computador de controle. Uma solução melhor é levar o sinal analógico para um VCO, que gera um sinal digital cuja freqüência de saída se altera de acordo com a entrada analógica. Este sinal digital é transmitido para o computador e será muito menos afetado pelo ruído. Circuitos no computador e controle contarão os pulsos digitais (isto é, realizam a função de medir a freqüência) para produzir um valor digital equivalente à entrada analógica original.

Modulação Sigma/Delta

Um outro modo para representar a informação analógica na forma digital é denominado **modulação sigma/delta**. Um conversor A/D sigma/delta é um dispositivo com "sobreamostragem", que significa que ele efetivamente amostra a informação analógica mais freqüentemente do que a taxa mínima de amostragem. A taxa mínima de amostragem é duas vezes maior do que a freqüência mais alta do sinal analógico de entrada. A sobreamostragem fornece pontos de dados interpolados entre aqueles que seriam gerados pela taxa mínima de amostragem. A abordagem sigma/delta, como a tensão-freqüência, não produz um número de vários bits para cada amostra. Em vez disso, ela representa a tensão analógi-

ca variando a densidade de 1s lógico em uma seqüência de bits de dados seriais. Para representar as partes positivas da forma de onda, uma seqüência de bits com uma densidade alta de 1s é gerada pelo conversor A/D (por exemplo, 0111110111110111110111). Para representar as partes negativas, uma densidade menor de 1s (isto é, uma densidade maior de 0s) é gerada (por exemplo, 0001000100001000001 0). A Fig. 10-22 mostra a relação entre o sinal analógico e sua seqüência de bits equivalente.

Questões de Revisão

1. Como o conversor A/D de rampa digital crescente/decrescente melhora o conversor A/D de rampa digital?
2. Qual é o principal elemento de um conversor A/D tensão-freqüência?
3. Cite duas vantagens e uma desvantagem do conversor A/D de rampa dupla.
4. Relacione três tipos de A/D que não usam um conversor D/A.
5. Quantos bits de dados um modulador sigma/delta utiliza?

10-14 VOLTÍMETRO DIGITAL

Um voltímetro digital converte uma tensão analógica para sua representação em código BCD, que então é decodificada e apresentada em algum tipo de display. A Fig. 10-23 mostra um circuito de voltímetro digital de três dígitos que utiliza um conversor A/D de rampa digital (mostrado no interior das linhas pontilhadas). Três contadores BCD em cascata fornecem as entradas para um conversor D/A BCD de três dígitos que tem um tamanho de degrau de 10 mV e uma saída de fundo de escala de 9,99 V. Cada estágio contador BCD também está ligado em um registrador de qua-

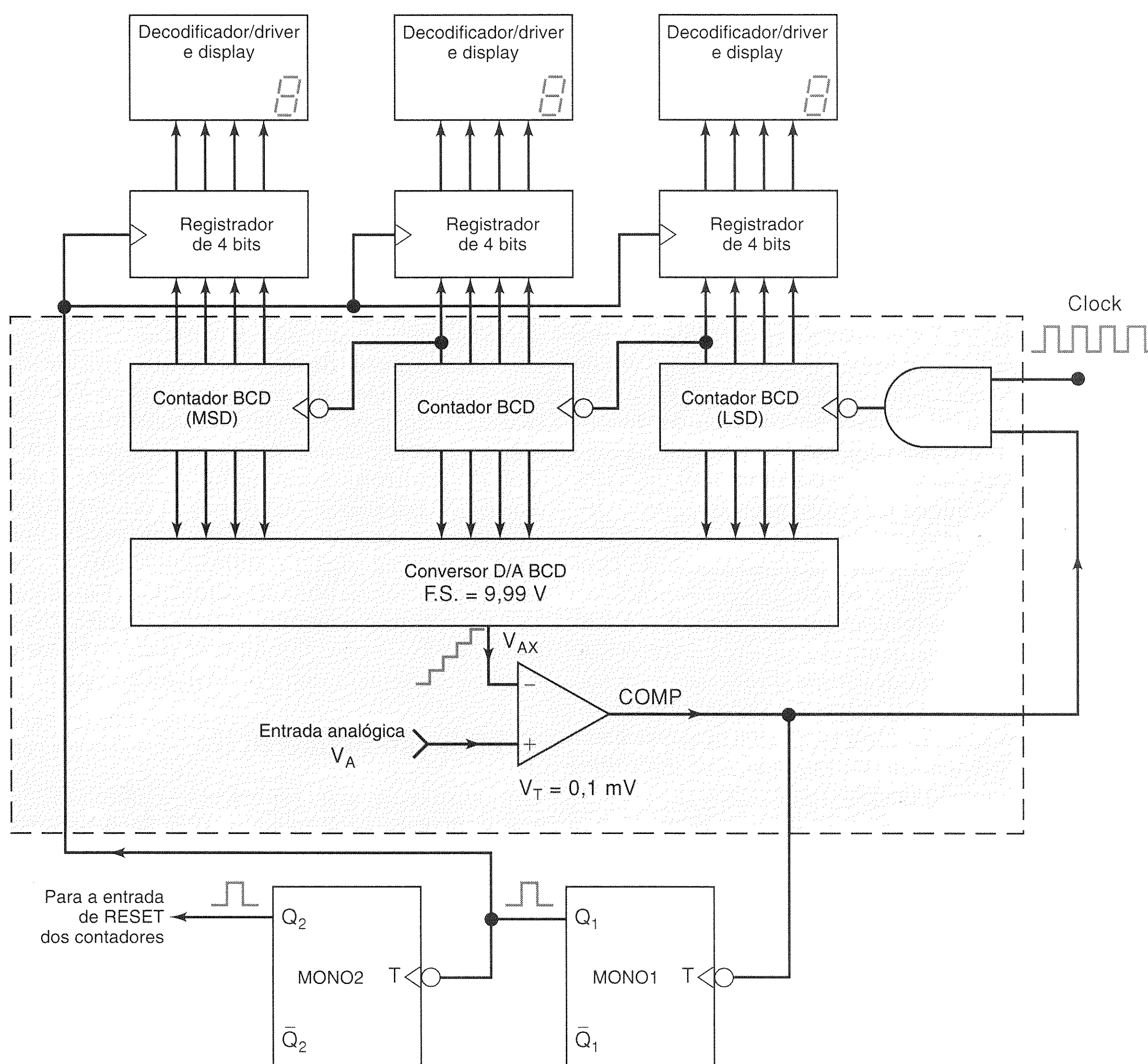


Fig. 10-23 Voltímetro digital de conversão contínua que utiliza um conversor A/D de rampa digital.

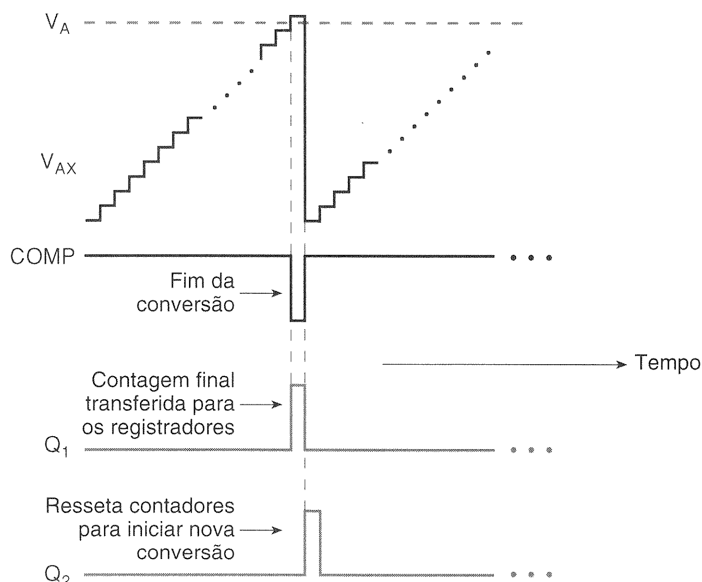


Fig. 10-24 Formas de onda para um voltímetro digital.

tro bits que aciona um decodificador/driver e um display. Os conteúdos dos contadores BCD são transferidos para os registradores no fim de cada ciclo de conversão, de modo que os displays não mostram os contadores ressetando e contando, mas apenas o final da contagem que representa a tensão desconhecida.

Enquanto $V_{AX} < V_A$, a saída COMP fica em ALTO, e pulsos de clock passam pela porta AND para o contador. Conforme o contador incrementa, o sinal V_{AX} aumenta a uma taxa de 10 mV por degrau e excede V_A (vide Fig. 10-24). Neste ponto, COMP vai para BAIXO para desabilitar a porta AND e parar a contagem, finalizando assim a conversão. A transição de descida de COMP também dispara o monoestável 1 (MONO1) para produzir um pulso de 1 μs em Q_1. A transição de subida de Q_1 transfere as saídas dos estágios contadores BCD para seus respectivos registradores para serem armazenadas e mostradas nos displays. A descida de Q_1 dispara um segundo monoestável, MONO2, para produzir um pulso que resseta todos os contadores para 0. Isto leva V_{AX} de volta para 0, e COMP retorna para ALTO permitindo pulsos no contador para o início de novo ciclo de conversão.

Assim, este voltímetro digital realizará uma conversão após a outra. Naturalmente, os registradores de armazenamento evitam que os displays mostrem o processo de conversão. Os valores dos displays mudarão apenas se V_A mudar, de modo que valores diferentes de contagem serão transferidos para os registradores no fim do ciclo de conversão.

Um exemplo numérico ajudará a ilustrar a operação deste circuito. Admita que V_A é 6,372 V. Para que a saída COMP comute para BAIXO, V_A deve exceder 6,3721 V. Já que a saída do conversor D/A aumenta de 10mV/degrau, isto requer

$$\frac{6{,}3721\ \text{V}}{10\ \text{mV}} = 637{,}21 \rightarrow 638\ \text{degraus}$$

Portanto, os contadores contarão até 638, que será transferido para os registradores e apresentado nos displays. Um pequeno LED pode ser usado para representar uma vírgula decimal de modo que o operador veja 6,38 V.

EXEMPLO 10-18

O que acontece se $V_A > 9{,}99$ V?

Solução

Com $V_A > 9{,}99$ V, a saída do comparador ficará em ALTO, permitindo que os pulsos de clock alcancem continuamente o contador. O contador repetidas vezes contará até 999 e reciclará para 000. O monoestável Q_1 *jamais* será disparado, e os conteúdos dos registradores manterão os valores anteriores. Um voltímetro digital bem-projetado deveria ter algum meio de detectar esta condição de *sobrefaixa* e ativar algum tipo de indicador. Um método possível seria simplesmente acrescentar um décimo terceiro bit ao contador. Este bit iria para ALTO somente se o contador reciclasse de 999 para 000, e então indicaria a condição de sobrefaixa.

O voltímetro digital pode ser modificado para mostrar tensões em várias faixas utilizando-se um amplificador ou atenuador adequado entre V_A e o comparador. Se V_A for 63,72 V, ela poderia ser atenuada por um fator 10, de modo que o comparador receberia 6,372 V na sua entrada + e os contadores mostrariam 638 nos displays no fim da conversão. O indicador da vírgula decimal seria colocado no LSD, de modo que os displays mostrariam 63,8 V.

O voltímetro digital pode ser modificado para operar como um multímetro que pode medir corrente e resistência, assim como tensão. Para medir corrente, a corrente desconhecida flui através de resistor fixo de referência para produzir uma tensão. Para medir resistência, uma corrente fixa de referência flui através de uma resistência desconhecida gerando uma tensão. Em cada caso, a tensão resultante é levada para um circuito como o da Fig. 10-23 para ser medida e apresentada nos displays. O circuito pode necessitar de um fator de atenuação de escala, de modo que o valor mostrado nos displays represente corretamente o que está sendo medido (corrente ou resistência).

Tensões AC podem ser medidas por este mesmo voltímetro digital se forem antes convertidas para tensões DC. Isto pode ser feito utilizando-se um circuito que converta o valor rms verdadeiro de um sinal (ac puro ou não) para um valor dc. O AD536A da Analog Devices é um CI que realiza esta conversão de rms para dc.

Questões de Revisão

1. Qual é o propósito dos registradores no circuito do voltímetro digital?
2. Qual é o objetivo dos dois monoestáveis?
3. Como a performance do voltímetro digital seria afetada se fosse utilizado um conversor A/D de aproximações sucessivas?

10-15 CIRCUITOS DE AMOSTRAGEM E RETENÇÃO (SAMPLE-AND-HOLD)

Quando uma tensão analógica é conectada diretamente na entrada de um conversor A/D, o processo de conversão pode ser afetado adversamente se a tensão analógica estiver mudando durante o processo de conversão. A estabilidade do processo de conversão pode ser melhorada utilizando-se um **circuito de amostragem e retenção (sample-and-hold circuit)** para manter a tensão analógica constante enquanto a conversão A/D se realiza. Um diagrama simplificado de um circuito de amostragem e retenção é mostrado na Fig. 10-25.

O circuito de amostragem e retenção contém um amplificador buffer de ganho unitário A_1 que apresenta uma alta impedância para o sinal analógico e tem uma baixa impedância de saída que pode carregar rapidamente o capacitor de retenção, C_h. O capacitor será conectado na saída de A_1 quando a chave digitalmente controlada for fechada. Isto é chamado *operação de amostragem*. A chave ficará fechada o suficiente para carregar o capacitor C_h com o valor atual da entrada analógica. Por exemplo, se a chave estiver fechada no instante t_0, a saída de A_1 carregará rapidamente C_h até a tensão V_0. Quando a chave abre, C_h *manterá* esta tensão, de modo que a saída de A_2 aplicará esta tensão ao conversor A/D. O amplificador buffer de ganho unitário A_2 apresenta uma alta impedância de entrada que não descarregará apreciavelmente a tensão do capacitor durante o tempo de conversão do A/D, e portanto o conversor A/D receberá essencialmente a tensão dc de entrada V_0.

Num sistema de aquisição de dados controlado por computador, como aquele apresentado anteriormente, a chave de amostragem e retenção seria controlada por um sinal digital gerado por um computador. O sinal do computador fecharia a chave para carregar C_h para uma nova amostra da tensão analógica; o intervalo de tempo que a chave teria que permanecer fechada é denominado **tempo de aquisição**, e ele depende do valor de C_h e das características do circuito de amostragem e retenção. O LF198 é um circuito de amostragem e retenção integrado que tem um tempo de aquisição típico de 4 μs para C_h = 1000 pF e 20 μs para C_h = 0,01 μF. O sinal do computador deveria abrir a chave para permitir que C_h mantenha seu valor e forneça uma tensão analógica relativamente constante na saída de A_2. Por exemplo, com o LF198, a tensão do capacitor descarregará tipicamente com uma taxa de apenas 30 mV por segundo com um capacitor de 1000 pF.

Questões de Revisão

1. Descreva a função de um circuito de amostragem e retenção.
2. *Verdadeiro ou falso:* Os amplificadores em um circuito de amostragem e retenção são usados para fornecer amplificação de tensão.

10-16 MULTIPLEXAÇÃO

Quando entradas analógicas de várias fontes devem ser convertidas, uma técnica de multiplexação pode ser utilizada para que o conversor A/D possa ser compartilhado. O esquema básico está ilustrado na Fig. 10-26 para um sistema de aquisição de quatro canais. A chave rotativa *S* é usada para selecionar cada sinal analógico para a entrada do conversor A/D, um de cada vez em seqüência. O circuito de controle controla a posição da chave de acordo com os bits do *endereço de seleção*, A_1, A_0, do contador de módulo quatro. Por exemplo, com A_1A_0 = 00, a chave conecta V_{A0} na entrada do conversor A/D; A_1A_0 = 01 conecta V_{A1} na entrada do conversor A/D; e assim por diante. Cada canal de entrada tem um código de endereço específico que, quando apresentado, conecta aquele canal no conversor A/D.

A operação se processa do seguinte modo:

1. Com o endereço de seleção = 00, V_{A0} é conectado na entrada do conversor A/D.
2. O circuito de controle gera um pulso de START para iniciar a conversão de V_{A0} para seu equivalente digital.

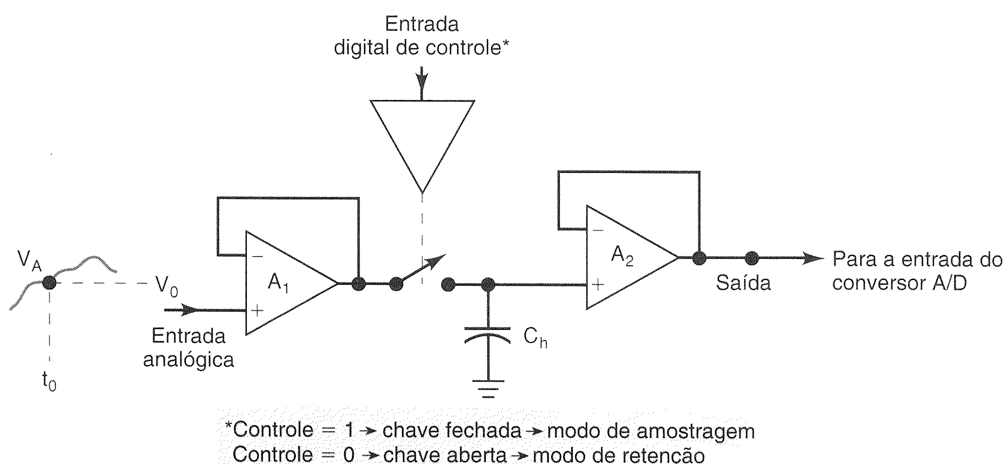


Fig. 10-25 Diagrama simplificado de um circuito de amostragem e retenção.

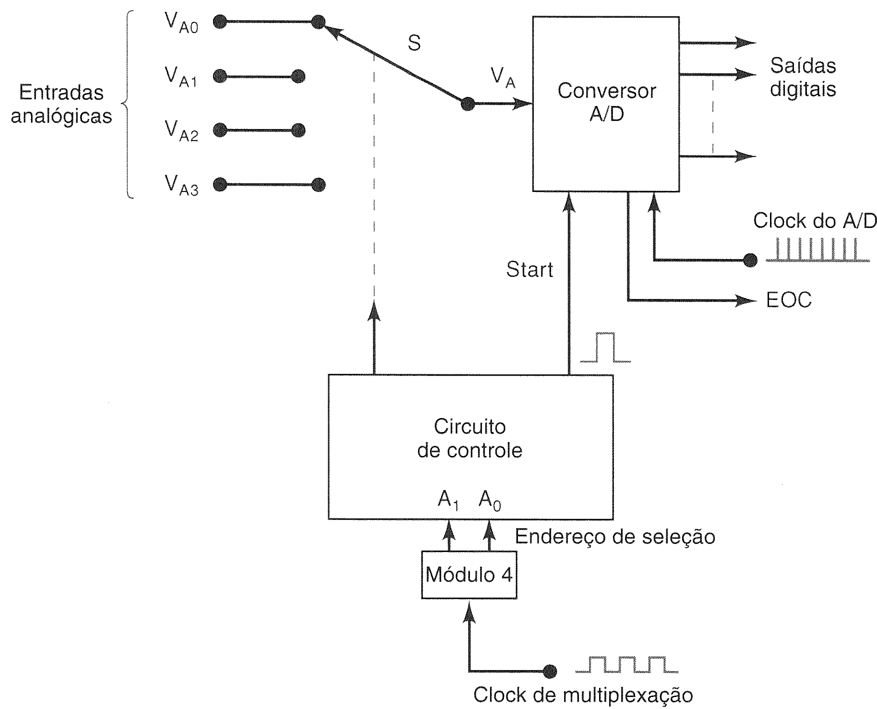


Fig. 10-26 Conversão de quatro entradas analógicas com multiplexação para um conversor A/D.

3. Quando a conversão está completa, *EOC* sinaliza que os dados de saída do conversor A/D estão prontos. Geralmente estes dados serão transferidos para um computador pelo barramento de dados.
4. O clock de multiplexação incrementa o endereço de seleção para 01, o que conecta V_{A1} no conversor A/D.
5. Os passos 2 e 3 são repetidos com o equivalente digital de V_{A1} presente nas saídas do conversor A/D.
6. O clock de multiplexação incrementa o endereço de seleção para 10, e V_{A2} é conectado no conversor A/D.
7. Os passos 2 e 3 são repetidos com o equivalente digital de V_{A2} presente nas saídas do conversor A/D.
8. O clock de multiplexação incrementa o endereço de seleção para 11, e V_{A3} é conectado no conversor A/D.
9. Os passos 2 e 3 são repetidos com o equivalente digital de V_{A3} presente nas saídas do conversor A/D.

O clock de multiplexação controla a taxa pela qual os sinais analógicos são seqüencialmente chaveados para o conversor A/D. A taxa máxima é determinada pelo tempo de atraso das chaves e pelo tempo de conversão do conversor A/D. O tempo de atraso das chaves pode ser minimizado utilizando-se chaves semicondutoras, tais como a chave bilateral CMOS descrita no Cap. 8. Pode ser necessário conectar um circuito de amostragem e retenção na entrada do conversor A/D se as entradas analógicas mudam significativamente durante o tempo de conversão do conversor A/D.

Vários conversores A/D integrados possuem os circuitos de multiplexação no mesmo chip do conversor. O ADC0808, por exemplo, pode multiplexar oito sinais analógicos de entrada diferentes para um conversor A/D. Ele utiliza um código de seleção de entrada de três bits para determinar qual das entradas analógicas é conectada ao conversor A/D.

Questões de Revisão

1. Qual é a vantagem deste esquema de multiplexação?
2. Como o contador de endereço deveria ser alterado se fossem oito entradas analógicas?

10-17 OSCILOSCÓPIO DE MEMÓRIA DIGITAL

Como um último exemplo de aplicação de conversores A/D e D/A, vamos analisar de maneira breve o osciloscópio de memória digital (osciloscópio digital). O osciloscópio digital utiliza estes dois dispositivos para digitalizar, armazenar e mostrar as formas de onda analógicas. O osciloscópio digital apresenta diversas vantagens sobre os osciloscópios de memória convencionais, que armazenam a imagem da forma de onda como cargas elétricas em uma malha de tela entre o canhão de elétrons do tubo de raios catódicos (CRT) e a tela revestida de fósforo. Aqui estão algumas destas vantagens:

- O osciloscópio digital pode armazenar formas de onda por um tempo indefinido, pois os valores digitais são armazenados em memória digital (isto é, flip-flops). Num osciloscópio de memória convencional, a imagem gradualmente desvanece conforme as cargas na malha de tela são lentamente neutralizadas.
- Em vários osciloscópios de memória digital, as formas de onda armazenadas podem ser posicionadas em qualquer lugar da tela, e suas escalas vertical e horizontal podem ser alteradas para se adaptar às condições de medida. Isto não é possível num osciloscópio de memória convencional.

- Com um osciloscópio de memória digital, é possível armazenar e mostrar partes da forma de onda que acontecem *antes* da ocorrência do ponto de disparo. Novamente, isto não pode ser feito com um osciloscópio de memória convencional.
- Muitos osciloscópios de memória digital podem armazenar várias formas de onda e imprimi-las numa impressora comum.

Um diagrama de blocos de um osciloscópio de memória digital é mostrado na Fig. 10-27. Toda a sua operação é controlada e sincronizada pelos circuitos do bloco de controle, que usualmente contém um microprocessador executando um programa de controle armazenado em ROM (*read-only memory*). A parte de aquisição de dados do sistema contém um circuito de amostragem e retenção (*sample-and-hold — S/H*) e um conversor A/D que repetidamente amostra e digitaliza o sinal de entrada numa taxa determinada pelo CLOCK DE AMOSTRAGEM e então envia o dado digitalizado para ser armazenado na memória. O bloco de CONTROLE garante que os dados correspondentes a pontos sucessivos são armazenados em posições de memória consecutivas atualizando continuamente o CONTADOR DE ENDEREÇOS da memória.

Quando a memória estiver cheia, o próximo dado do conversor A/D será armazenado na primeira posição de memória, escrevendo sobre o dado anterior, e assim por diante, para os pontos sucessivos. Este processo de aquisição e armazenamento de dados continua até que o bloco de controle receba um sinal de disparo, que pode ser gerado a partir da própria forma de onda (disparo INTERNO) ou ser fornecido por uma fonte de disparo externo. Quando ocorre o disparo, o sistema pára de adquirir novos dados e entra no modo display, no qual uma parte ou toda a memória é repetidamente apresentada no CRT.

A operação de apresentação dos dados no display utiliza dois conversores D/A para fornecerem as tensões de deflexão vertical e horizontal para o CRT. Os dados da memória produzem a deflexão vertical do feixe de elétrons, enquanto o CONTADOR DE BASE DE TEMPO aciona a deflexão horizontal com um sinal de varredura do tipo escada. O bloco de CONTROLE sincroniza a operação do display incrementando ao mesmo tempo o CONTADOR DE ENDEREÇOS da memória e o CONTADOR DE BASE DE TEMPO, de modo que cada passo horizontal do feixe de elétrons é acompanhado por um dado novo da memória para o conversor D/A vertical. Os contadores reciclam continuamente, de modo que os valores dos dados correspondentes aos pontos adquiridos são repetidamente apresentados na tela do CRT. A imagem na tela é formada por pequenos pontos que representam os diversos dados obtidos, mas o número de pontos normalmente é tão grande (geralmente 1000 ou mais) que eles tendem a se agrupar e parecem ser uma forma de onda suave e contínua. A operação de apresentação de dados no display termina quando o operador pressiona um botão frontal que comanda o osciloscópio de memória digital a iniciar um novo ciclo de aquisição de dados.

Aplicações Relacionadas

A mesma seqüência de operações realizada num osciloscópio de memória digital — aquisição de dados, digitalização, armazenamento e apresentação — é utilizada em outras aplicações de conversores A/D e D/A. Por exemplo, em gravação digital de áudio, o sinal analógico de áudio produzido por um microfone é digitalizado (utilizando um conversor A/D) e então armazenado em algum meio, tais como fita magnética, disco magnético ou ótico. Posteriormente, os dados armazenados são "reproduzidos" enviando-os para um conversor D/A para reconstruir o sinal analógico, que é levado para o sistema de amplificador e alto-falantes para produzir o som gravado.

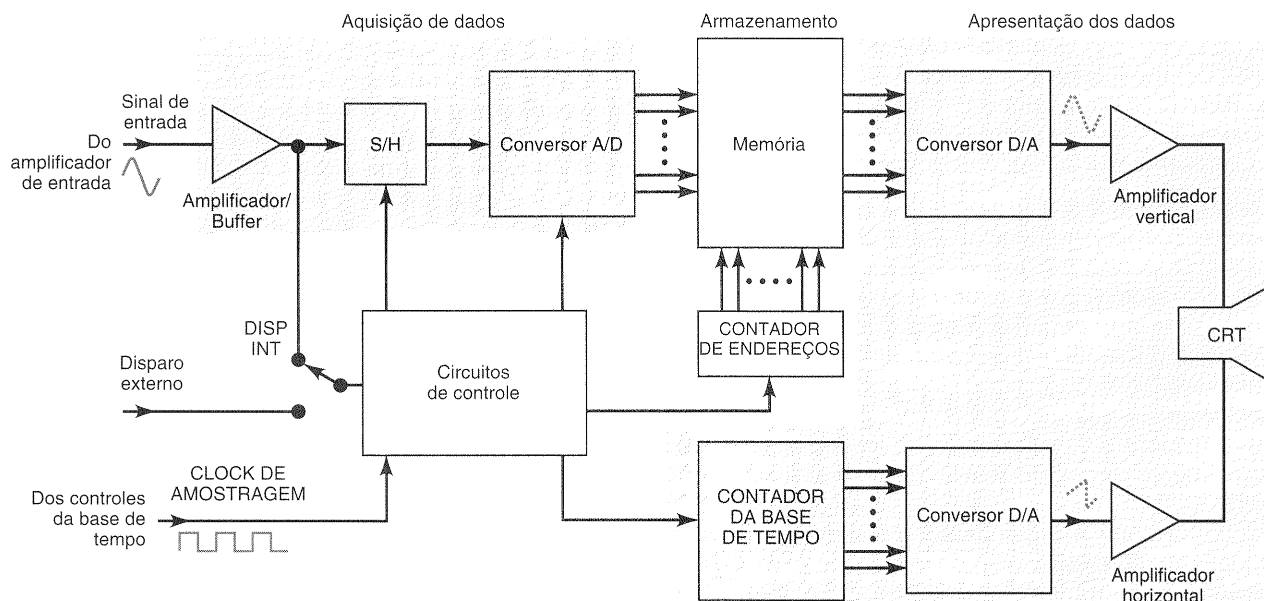


Fig. 10-27 Diagrama de blocos de um osciloscópio de memória digital.

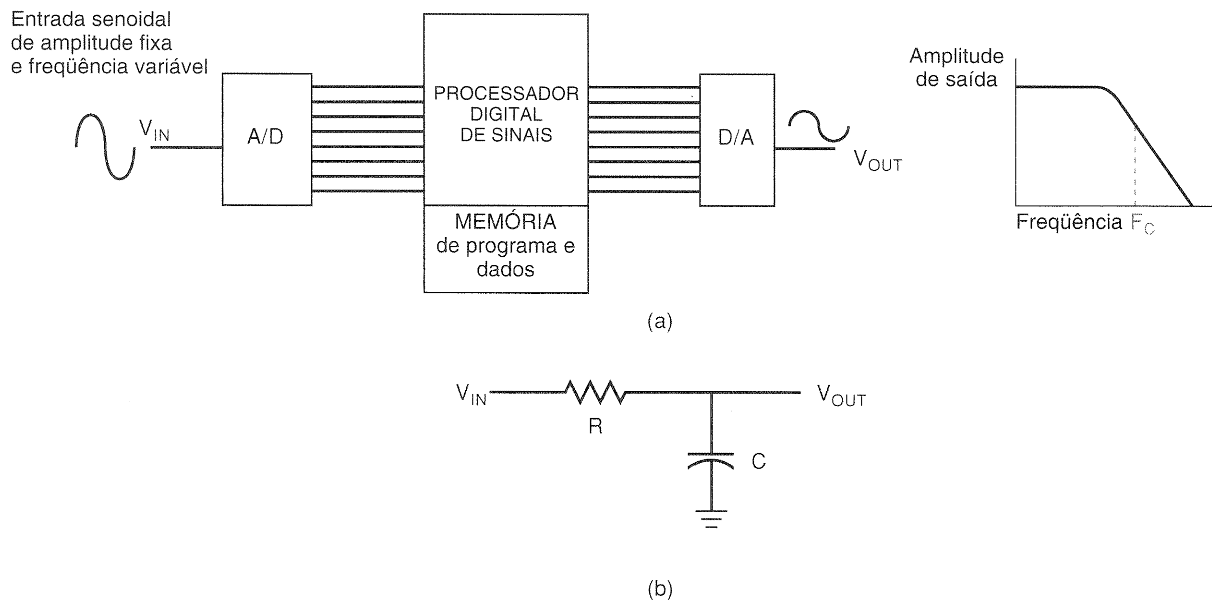


Fig. 10-28 (a) Filtragem passa-baixa realizada por DSP; (b) um filtro passa-baixa RC simples.

Questões de Revisão

1. Cite algumas vantagens de um osciloscópio de memória digital quando comparado com um osciloscópio de memória convencional.
2. Descreva as funções dos conversores A/D e D/A que fazem parte de um osciloscópio de memória digital.

10-18 PROCESSAMENTO DIGITAL DE SINAIS (DSP)

Uma das áreas de sistemas digitais mais dinâmicas hoje em dia está no campo de **processamento digital de sinais** ou **DSP** (*digital signal processing*). Um processador DSP é uma forma muito especializada de microprocessador que foi otimizado para realizar cálculos repetitivos sobre uma série de dados digitalizados. Os dados digitalizados normalmente vão sendo enviados por um conversor A/D. Está fora do escopo deste livro explicar a matemática que permite ao DSP o processamento destes valores, mas basta dizer que, para cada novo valor que recebe, cálculos são realizados (muito rapidamente). Estes cálculos envolvem o dado mais recente, assim como muitas amostras anteriores. O resultado do cálculo produz um novo dado de saída, que normalmente é enviado para um conversor D/A. Um diagrama de blocos de um sistema deste tipo é mostrado na Fig. 10-28(a).

A principal aplicação de DSP está em filtragem e condicionamento de sinais analógicos. Como um exemplo bastante simples, um DSP pode ser programado para receber uma forma de onda, tal como a saída de um pré-amplificador de áudio e passar para a saída os componentes de freqüência que estiverem abaixo de uma certa freqüência. Todas as freqüências mais altas são atenuadas pelo filtro. Você se lembra, dos seus estudos sobre circuitos analógicos, que a mesma coisa pode ser conseguida por um simples filtro passa-baixa construído com um capacitor e um resistor, conforme mostrado na Fig. 10-28(b). A vantagem do DSP sobre resistores e capacitores é a flexibilidade de ele ser capaz de alterar a freqüência de corte sem nenhuma substituição de componentes. Em vez disso, os cálculos são alterados para adaptar a resposta dinâmica do filtro. Você já esteve num auditório no qual o sistema de som começou a apitar? Isto pode ser evitado se a freqüência crítica realimentada puder ser filtrada. Infelizmente, a freqüência que causa o apito se altera com o número de pessoas na sala, com as roupas que estiverem usando e com muitos outros fatores. Com um equalizador de áudio baseado em DSP, a freqüência de oscilação pode ser detectada e o filtro dinamicamente ajustado para sintonizá-la.

DSP está sendo integrado em diversos sistemas comuns que são familiares a você. Em aparelhos de CD, por exemplo, DSP é utilizado para implementar um filtro digital de ruídos. Os dados saem do CD numa sucessão de dados seriais e são decodificados para um formato digital padrão *inter-IC sound* (I^2S). As palavras de dados no formato I^2S são levadas para um circuito DSP que realiza filtragem com interpolação e sobreamostragem que desloca as componentes indesejadas de ruído para uma faixa de freqüência mais alta, que pode ser facilmente removida por filtros analógicos. A saída do filtro digital é enviada para o conversor D/A. Os dados podem estar em paralelo para um D/A baseado em divisão de corrente (Fig. 10-9) ou podem ser uma forma de onda digital sigma/delta modulada para um conversor D/A sigma/delta. O conversor D/A, por sua vez, transforma esta informação digital numa forma de onda analógica, que através de filtragem analógica tem suas bordas suavizadas atenuando-se as freqüências mais altas indesejáveis, e produz um sinal de áudio quase exatamente igual ao original.

Aplicações baseadas em DSP estão crescendo agora na mesma taxa que as aplicações de microprocessadores cresceram no começo dos anos 1980. Elas fornecem uma solução digital para muitos problemas analógicos tradicionais.

Alguns exemplos de aplicações são efeitos especiais de áudio, reconhecimento de voz, criptografia de dados para telecomunicações, cancelamento de eco, Transformada Rápida de Fourier, processamento de imagens em televisão digital, geração de feixes em eletrônica biomédica e cancelamento de ruído em controles industriais. Conforme esta tendência continua, podemos esperar ver em breve praticamente todos os sistemas eletrônicos contendo circuitos de processamento digital de sinais.

Questões de Revisão

1. Qual é a principal aplicação de DSP?
2. Qual é a fonte típica de dados digitais para o DSP processar?
3. Que vantagem apresenta um filtro DSP sobre um filtro analógico?

RESUMO

1. Variáveis físicas que desejamos medir, tais como temperatura, pressão, umidade, distância e velocidade, entre outras, são quantidades que variam continuamente. Um transdutor pode ser utilizado para transformar estas quantidades em sinais elétricos de tensão ou corrente que são proporcionais às variáveis físicas. Estes sinais de corrente ou tensão que variam continuamente são denominados sinais *analógicos*.
2. Para medir uma variável física, um sistema digital deve atribuir um número binário ao valor analógico que está presente naquele instante. Isto é executado por um conversor A/D. Para gerar valores de tensões ou correntes variáveis que possam controlar um processo físico, um sistema digital deve transformar números binários em tensões e correntes. Isto é realizado por um conversor D/A.
3. Um conversor D/A de n bits divide uma faixa de valores analógicos (tensão ou corrente) em $2^n - 1$ partes. O tamanho de cada pedaço é o peso analógico correspondente ao dígito menos significativo. Isto é chamado de *resolução* ou *tamanho do degrau*.
4. A maioria dos conversores D/A utiliza redes resistivas que fazem quantidades ponderadas de corrente fluírem quando alguma das entradas binárias é ativada. A quantidade de corrente que flui é proporcional ao peso binário de cada bit de entrada. Estas correntes ponderadas são somadas para gerar o sinal analógico de saída.
5. Um conversor A/D deve atribuir um número binário a uma quantidade analógica (continuamente variável). A exatidão com a qual um conversor A/D pode realizar esta conversão é dependente da quantidade de números que ele pode atribuir e do tamanho da faixa analógica. A menor alteração do valor analógico que um A/D pode medir é chamada de *resolução*, o peso do seu bit menos significativo.
6. Amostrando repetidamente o sinal analógico de entrada, convertendo-o para digital e armazenando os valores digitais numa memória, uma forma de onda analógica pode ser capturada. Para reconstruir o sinal, os valores digitais são lidos da memória na mesma taxa com a qual foram armazenados e então levados para um conversor D/A. A saída do D/A é filtrada para suavizar os degraus da escada e recriar a forma de onda original.
7. Um conversor A/D de rampa digital é o mais simples de compreender mas não é utilizado com freqüência devido ao seu tempo de conversão variável. Um conversor de aproximações sucessivas apresenta um tempo constante de conversão e provavelmente é o conversor de propósito geral mais comum.
8. Conversores do tipo flash usam comparadores analógicos e um codificador de prioridade para atribuírem um valor digital para uma entrada analógica. Estes são os conversores mais rápidos, pois os únicos atrasos envolvidos são os atrasos de propagação.
9. Outros métodos populares para conversão A/D são: rastreador crescente/decrescente, rampa dupla, conversão tensão-freqüência e conversão sigma/delta. Cada tipo de conversor tem seu próprio nicho de aplicações.
10. Qualquer conversor A/D pode ser utilizado com outros circuitos, tais como os multiplexadores analógicos que selecionam um entre vários sinais analógicos para ser convertido, um de cada vez. Circuitos de amostragem e retenção (sample-and-hold) podem ser usados para "congelar" um sinal analógico que muda rapidamente enquanto a conversão é realizada.
11. Processamento digital de sinais é um novo campo fascinante e de grande evolução na eletrônica. Dispositivos para DSP permitem que cálculos sejam realizados rapidamente para emular digitalmente a operação de muitos circuitos de filtros analógicos. DSP é responsável por muitos dos recentes avanços em áudio de alta fidelidade, TV de alta definição e nas telecomunicações, propiciando métodos para eliminação de ruído randômico, crosstalk e interferências sem afetar adversamente o sinal.

TERMOS IMPORTANTES

quantidade digital
quantidade analógica
transdutor
conversor analógico-digital (A/D)
conversor digital-analógico (D/A)
saída de fundo de escala
tamanho do degrau/resolução
escada
saída de fundo de escala
erro de fundo de escala
erro de linearidade
erro de offset
tempo de estabilização
monotonicidade
digitalização
conversor A/D de rampa digital
erro de quantização
conversão A/D de aproximações sucessivas
entradas diferenciais
entrada de ESCRITA
conversor A/D flash
conversão A/D de rampa digital crescente/decrescente
conversão A/D de rampa dupla
conversão A/D tensão-freqüência
modulação sigma/delta
circuito de amostragem e retenção (sample-and-hold)
tempo de aquisição
processamento digital de sinais (DSP)

PROBLEMAS

SEÇÕES 10-1 E 10-2

10-1. QUESTÃO DE FIXAÇÃO

(a) Qual é a expressão que relaciona a saída com as entradas de um conversor D/A?

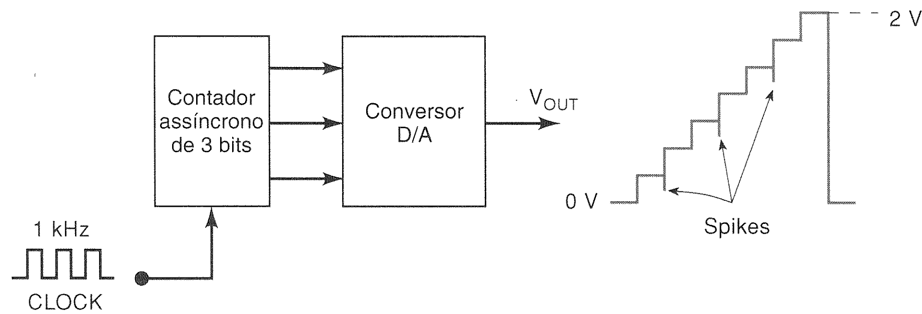


Fig. 10-29 Problema 10-8.

(b) Defina *tamanho do degrau* para um conversor D/A.
(c) Defina *resolução* de um conversor D/A.
(d) Defina *fundo de escala*.
(e) Defina *resolução percentual*.
(f) *Verdadeiro ou falso:* Um conversor D/A de 10 bits terá uma resolução menor do que um conversor D/A de 12 bits para a mesma saída de fundo de escala.
(g) *Verdadeiro ou falso:* Um conversor D/A de 10 bits com saída de fundo de escala de 10 V tem uma resolução percentual menor do que um conversor D/A de 10 bits com fundo de escala de 12 V.

10-2. Um conversor D/A de oito bits produz uma tensão de saída de 2 V para um código de entrada de 01100100. Qual será o valor de V_{OUT} para um código de entrada de 10110011?

10-3. Determine o peso de cada bit de entrada do conversor D/A do Problema 10-2.

10-4. Qual é a resolução do conversor D/A do Problema 10-2? Expresse-a em volts e em porcentagem.

10-5. Qual é a resolução em volts de um conversor D/A de 10 bits cuja saída de F.S. é 5 V?

10-6. Quantos bits são necessários para um conversor D/A ter uma saída de F.S. de 10 mA com resolução menor do que 40 μA?

10-7. Qual é a resolução percentual do conversor D/A da Fig. 10-29? Qual é o tamanho do degrau se o degrau superior é 2 V?

C

10-8. Qual é a causa dos spikes na forma de onda de V_{OUT} da Fig. 10-29? (*Sugestão:* Note que o contador é assíncrono e que os spikes ocorrem em degraus alternados.)

10-9. Admitindo um conversor D/A de 12 bits com precisão perfeita, quão próxima de 250 rpm a velocidade do motor pode ser ajustada na Fig. 10-4?

10-10. Um conversor D/A de 12 bits que utiliza código de entrada BCD (três dígitos) tem uma saída de fundo de escala de 9,99 V. Determine o tamanho do degrau, a resolução percentual e o valor de V_{OUT} para um código de entrada de 011010010101.

10-11. Compare o tamanho de um degrau e a resolução percentual de um conversor D/A com uma entrada binária de oito bits com o conversor D/A que tem uma entrada BCD de oito bits. Considere que o fundo de escala é 990 mV para cada um dos conversores.

10-12. Determine o peso de cada bit para o conversor D/A BCD do Problema 10-10. Qual você acha que seria a saída se a entrada fosse 100000100011?

SEÇÃO 10-3

D

10-13. O tamanho do degrau do conversor D/A da Fig. 10-6 pode ser alterado mudando-se o valor de R_F. Determine o valor de R_F necessário para um tamanho de degrau de 0,5 V. O novo valor de R_F vai alterar a resolução percentual?

D

10-14. Suponha que a saída do conversor D/A da Fig. 10-8(a) está conectada ao amplificador operacional da Fig. 10-8(b).
(a) Com $V_{REF} = 5$ V, $R = 20$ kΩ e $R_F = 10$ kΩ, determine o tamanho do degrau e a tensão de fundo de escala de V_{OUT}.
(b) Altere o valor de R_F de modo que a tensão de fundo de escala de V_{OUT} seja -2 V.
(c) Utilize este novo valor de R_F e determine o fator de proporcionalidade, K, na relação $V_{OUT} = K(V_{REF} \times B)$.

10-15. Qual é a vantagem do conversor D/A da Fig. 10-9 sobre o da Fig. 10-8, especialmente para um grande número de bits de entrada?

SEÇÕES 10-4 A 10-6

10-16. Um conversor D/A de oito bits tem um erro de fundo de escala de 0,2 % F.S. Se o conversor D/A tem uma saída de fundo de escala de 10 mA, qual é o máximo que ele pode apresentar de erro para qualquer entrada digital? Se a saída do D/A indica 50 μA para uma entrada digital de 00000001, isto está dentro da faixa de precisão especificada? (Admita que não existe erro de offset.)

C, N

10-17. O controle de um dispositivo de posicionamento pode ser feito utilizando-se um *servomotor*, que é um motor projetado para acionar um dispositivo mecânico enquanto existir um sinal de erro. A Fig. 10-30 mostra um sistema servocontrolado simples que é controlado por uma entrada digital que poderia estar vindo diretamente de um computador ou de um meio de saída, tal como uma fita magnética. A alavanca é movida verticalmente pelo servomotor. O motor gira no sentido horário ou anti-horário, dependendo de a tensão do amplificador de potência (A.P.) ser positiva ou negativa. O motor pára quando a saída do A.P. é 0.

A posição mecânica da alavanca é convertida para uma tensão dc por um potenciômetro acoplado a ela. Quando a alavanca está no seu ponto de referência 0, $V_P = 0$ V. O valor de V_P aumenta a uma taxa de 1 V/centímetro até que a alavanca alcance seu ponto mais alto (10 centímetros) e $V_P = 10$ V. A posição desejada da alavanca é fornecida por um código digital do computador que é levado para o conversor D/A, produzindo V_A. A *diferença* entre V_P e V_A (chamada de *erro*) é produzida pelo amplificador *diferencial* e é amplificada por A.P. para acionar o motor no sentido que cause a diminuição do sinal de erro para 0, isto é, movimenta a alavanca até que $V_P = V_A$.

(a) Para posicionar a alavanca com uma resolução de 0,1 cm, quantos bits seriam necessários no código digital de entrada?

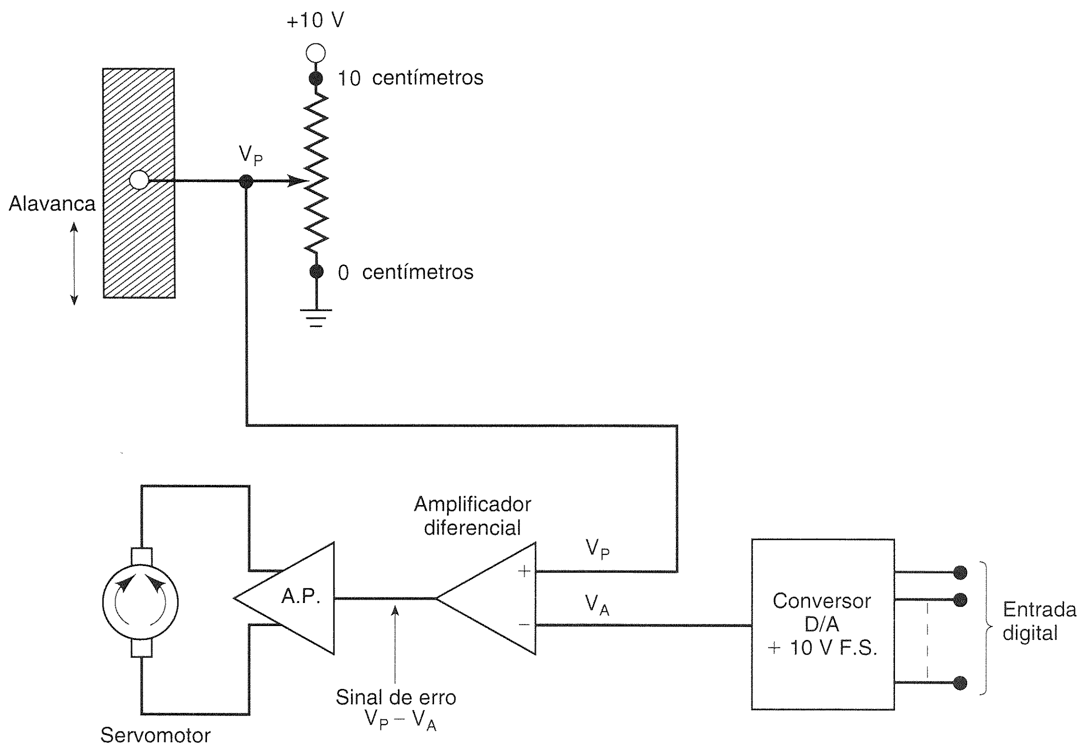


Fig. 10-30 Problema 10-17.

(b) Na operação real, a alavanca pode oscilar ligeiramente em torno da posição desejada, especialmente se um potenciômetro de *fio* for utilizado. Você pode explicar o porquê?

10-18. QUESTÃO DE FIXAÇÃO

(a) Defina *rede de resistores binariamente ponderados*.

(b) Defina *rede R/2R*.

(c) Defina *tempo de estabilização* de um conversor D/A.

(d) Defina *erro de fundo de escala*.

(e) Defina *erro de offset*.

10-19. Um determinado conversor D/A de seis bits tem uma saída com fundo de escala de 1,260 V. Sua precisão está especificada como ± 0,1% F.S., e ele tem um erro de offset de ± 1 mV. Admita que o erro de offset não possa ser zerado. Considere as medições feitas neste D/A (Tabela 10-7) e determine quais delas não estão dentro das especificações do dispositivo. (*Sugestão:* O erro de offset é somado ao erro causado pelas imprecisões dos componentes.)

SEÇÃO 10-7

T

10-20. Um certo conversor D/A tem as seguintes especificações: oito bits de resolução, fundo de escala = 2,55 V, offset ≤ 2 mV; precisão = ±0,1% F.S. Um teste estático neste conversor D/A produziu os resultados mostrados na Tabela 10-8. Qual é a provável causa do mau funcionamento?

TABELA 10-7

Código de Entrada	Saída
000010	41,5 mV
000111	140,2 mV
001100	242,5 mV
111111	1,258 V

TABELA 10-8

Código de Entrada	Saída
00000000	8 mV
00000001	18,2 mV
00000010	28,5 mV
00000100	48,3 mV
00001111	158,3 mV
10000000	1,289 V

T

10-21. Repita o Problema 10-20 utilizando os valores medidos apresentados na Tabela 10-9.

TABELA 10-9

Código de Entrada	Saída
00000000	20,5 mV
00000001	30,5 mV
00000010	20,5 mV
00000100	60,6 mV
00001111	150,6 mV
10000000	1,300 V

T

10-22. Um estudante conecta um contador no conversor D/A da Fig. 10-3 para realizar o teste da forma de onda do tipo escada utilizando um clock de 1 kHz. O resultado é mostrado na Fig. 10-31. Qual é a causa provável para este sinal tipo escada estar incorreto?

SEÇÕES 10-8 E 10-9

10-23. QUESTÃO DE FIXAÇÃO

Preencha os espaços na seguinte descrição do conversor A/D da Fig. 10-13. Cada espaço pode ter uma ou mais palavras.

Um pulso de START é aplicado para _____ o contador e para impedir o ______ de passar através da porta AND para o _____. Neste ponto, a saída do conversor D/A, V_{AX}, é _____ e $\overline{EOC}$ é _____.

Quando START retorna a ______, a porta AND é _____, e o contador é habilitado a _____. O sinal V_{AX} é aumentado um _____ de cada vez até que ele _____ V_A. Neste ponto, ____vai para BAIXO para _____pulsos de _____. Isto sinaliza o final da conversão e o equivalente digital de V_A está presente na _______.

10-24. Um conversor A/D de rampa digital de oito bits com uma resolução de 40 mV utiliza uma freqüência de clock de 2,5 MHz e um comparador com V_T = 1 mV. Determine os seguintes valores:

(a) A saída digital para V_A = 6,000 V
(b) A saída digital para 6,035 V
(c) Os tempos de conversão máximo e médio para este conversor A/D

10-25. Por que as saídas digitais para os itens (a) e (b) do Problema 10-24 são iguais?

D

10-26. O que aconteceria no conversor A/D do Problema 10-24 se uma tensão analógica de V_A = 10,853 V fosse aplicada na entrada? Que forma de onda apareceria na saída do D/A? Acrescente a lógica necessária a este A/D, de modo que uma indicação de "fora de escala" fosse gerada sempre que V_A fosse muito grande.

10-27. Um conversor A/D tem as seguintes características: resolução de 12 bits; erro de fundo de escala de 0,03% F.S.; fundo de escala de +5 V.

(a) Qual é o erro de quantização em volts?
(b) Qual é o erro total possível em volts?

C, N

10-28. O erro de quantização de um conversor A/D como o da Fig. 10-13 sempre é positivo, pois o valor de V_{AX} deve exceder V_A para a saída do comparador mudar de estado. Isto significa que o valor de V_{AX} poderia ser até 1 LSB maior do que V_A. Este erro de quantização pode ser modificado de modo que V_{AX} fique dentro de ±1/2 LSB de V_A. Isto pode ser feito adicionando-se uma tensão fixa igual a 1/2 LSB (1/2 degrau) ao valor de V_A. A Fig. 10-32 mostra isto simbolicamente para um conversor que tem uma resolução de 10 mV/degrau. Uma tensão fixa de +5 mV é somada com a saída do conversor D/A pelo amplificador somador, e o resultado, V_{AY}, é levado para o comparador, que tem V_T = 1 mV.

Para este conversor modificado, determine a saída digital para os seguintes valores de V_A:

(a) V_A = 5,022 V
(b) V_A = 50,28 V

Determine o erro de quantização em cada caso comparando V_{AX} e V_A. Repare que o erro é positivo num caso e negativo no outro.

C

10-29. Para o conversor A/D da Fig. 10-32, determine a faixa de valores analógicos de entrada que produzirão uma saída digital de 0100011100.

SEÇÃO 10-10

N

10-30. Admita que o sinal analógico na Fig. 10-33(a) deve ser digitalizado realizando-se conversões A/D contínuas utilizando um conversor de rampa digital de oito bits cuja rampa cresce numa taxa de 1 V a cada 25 μs. Esboce o sinal reconstruído utilizando os dados obtidos durante o processo de digitalização. Compare-o com o sinal original e discuta o que poderia ser feito para torná-lo uma representação mais precisa.

N, C

10-31. Se uma freqüência mais alta do que metade da taxa de amostragem é aplicada num conversor A/D, uma forma de onda errada de baixa freqüência será reconstituída com freqüência igual à diferença entre a freqüência de amostragem (amostras por segundo) e a freqüência de entrada (ciclos por segundo). Este fenômeno é chama-

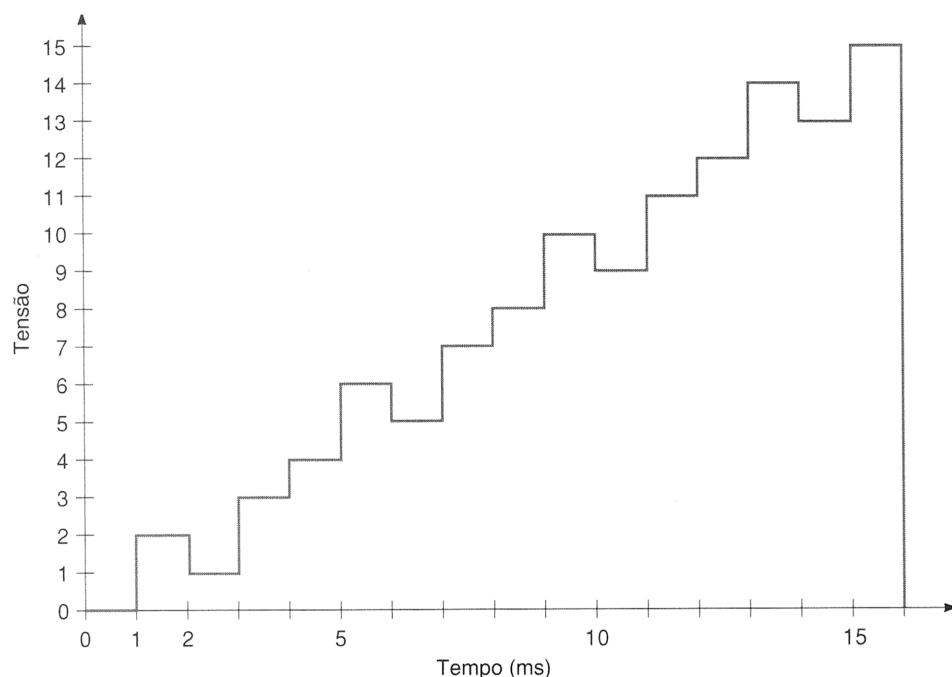


Fig. 10-31 Problema 10-22.

Fig. 10-32 Problemas 10-28 e 10-29.

do *aliasing* e merece grande atenção quando se amostra um sinal analógico. Na senóide da Fig. 10-33(b), marque os pontos relativos às amostras feitas pelo conversor A/D flash em intervalos de 75 μs (começando na origem). Então, desenhe a saída reconstruída a partir do conversor D/A (conecte os pontos de amostra com linhas retas para mostrar a filtragem). Calcule a freqüência de amostragem, a freqüência da entrada senoidal e a diferença entre elas. Compare com a freqüência da forma de onda reconstruída.

SEÇÃO 10-11

10-32. QUESTÃO DE FIXAÇÃO
Indique, para cada uma das seqüências a seguir, se ela se refere a um conversor A/D de rampa digital ou a um de aproximações sucessivas, ou a ambos.
- **(a)** Produz uma forma de onda do tipo escada na saída do conversor D/A
- **(b)** Tem um tempo de conversão constante independente de V_A
- **(c)** Tem um tempo de conversão médio mais curto
- **(d)** Utiliza um comparador analógico
- **(e)** Utiliza um conversor D/A
- **(f)** Utiliza um contador
- **(g)** Tem uma complexa lógica de controle
- **(h)** Tem uma saída $\overline{EOC}$

10-33. Desenhe a forma de onda para V_{AX} conforme o conversor A/D de aproximações sucessivas da Fig 10-18 converte V_A = 6,7 V.

10-34. Repita o Problema 10-33 para V_A = 16 V.

10-35. Um conversor de aproximações sucessivas de oito bits tem um fundo de escala de 2,55 V. O tempo de conversão para V_A = 1 V é 80 μs. Qual será o tempo de conversão para V_A = 1,5 V?

10-36. A Fig. 10-34 mostra a forma de onda V_{AX} durante um ciclo completo de conversão para um conversor A/D de aproximações sucessivas de seis bits com um tamanho de degrau de 40 mV. Examine esta forma de onda e descreva o que está ocorrendo nos instantes t_0 a t_5. Determine, então, a saída digital resultante.

10-37. Consulte a Fig. 10-20. Qual é o valor aproximado da entrada analógica se o barramento de dados do microcomputador apresenta 10010111 quando $\overline{RD}$ é pulsado em BAIXO?

D

10-38. Conecte uma fonte de referência de 2,0 V em $V_{ref}/2$ e repita o Problema 10-37.

C, D

10-39. Projete uma interface A/D para um termostato digital americano utilizando um sensor de temperatura LM34 e o ADC0804. Seu sistema deve medir com precisão (±0,2°F) de 50 até 101°F. O LM34 fornece 0,01 V por grau F (0°F = 0 V).
- **(a)** Qual deveria ser o valor digital para 50°F para a melhor resolução?
- **(b)** Que tensão deve ser aplicada a $V_{in}(-)$?
- **(c)** Qual é a faixa do fundo de escala da tensão na entrada?
- **(d)** Que tensão deve ser aplicada em $V_{ref}/2$?
- **(e)** Que valor binário representará 72°F?
- **(f)** Qual é a resolução em °F? E em volts?

SEÇÃO 10-12

10-40. Discuta como um conversor A/D flash com um tempo de conversão de 1 μs funcionaria para a situação do Problema 10-30.

D

10-41. Desenhe o diagrama do circuito para um conversor flash de quatro bits com saída BCD e resolução de 0,1 V. Considere que uma fonte de alimentação precisa de +5 V está disponível.

10-42. QUESTÃO DE FIXAÇÃO
Para cada uma das sentenças a seguir, indique o tipo de conversor A/D que está sendo descrito: rampa digital, aproximações sucessivas ou flash.

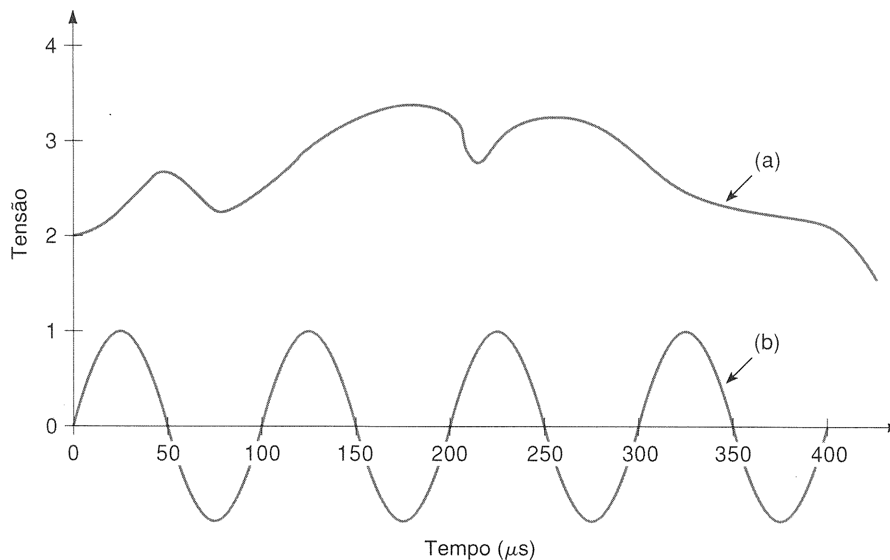


Fig. 10-33 Problemas 10-30, 10-31 e 10-40.

(a) Método de conversão mais rápido
(b) Necessita de um pulso de START
(c) Requer mais circuito
(d) Não utiliza um conversor D/A
(e) Gera uma forma de onda do tipo escada
(f) Utiliza um comparador analógico
(g) Tem um tempo de conversão relativamente fixo independente de V_A

SEÇÃO 10-13

10-43. QUESTÃO DE FIXAÇÃO
Para cada uma das sentenças a seguir, indique o(s) tipo(s) de conversor(es) A/D que está(ão) sendo descrito(s).
(a) Utiliza um contador que nunca é ressetado para 0
(b) Utiliza um grande número de comparadores
(c) Utiliza um VCO
(d) É utilizado nos ambientes industriais ruidosos
(e) Utiliza um capacitor
(f) É relativamente insensível à temperatura

SEÇÃO 10-14

10-44. Uma tensão V_A = 3,853 V é aplicada ao voltímetro digital da Fig. 10-23. Admita que os contadores estão inicialmente ressetados em 0 e esboce as formas de onda em V_{AX}, COMP, Q_1, Q_2 e na saída da porta AND durante *dois* ciclos completos. Utilize um clock de 100 kHz e considere que os monoestáveis têm um t_P = 1 μs.

T
10-45. Consulte o circuito do voltímetro digital da Fig. 10-23. Admita que o circuito funciona corretamente para valores de V_A até aproximadamente 7,99 V. Entretanto, quando V_A ultrapassa 8 V, os displays não alteram a leitura anterior, a forma de onda do tipo escada em V_{AX} vai até aproximadamente 7,99 V antes de retornar para 0, e COMP permanece em ALTO. Quais são algumas das possíveis causas do mau funcionamento?

D
10-46. Acrescente a lógica necessária ao voltímetro digital da Fig. 10-23 para acionar um LED de fora de escala sempre que

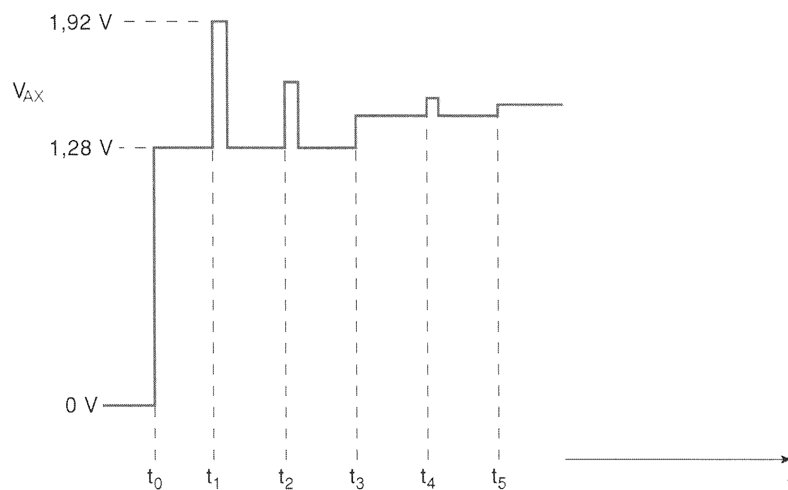


Fig. 10-34 Problema 10-36.

a tensão de entrada analógica exceder 9,99 V. O LED deve ser desligado no início de cada nova conversão.

SEÇÕES 10-15 E 10-16

T

10-47. Consulte o circuito de amostragem e retenção da Fig. 10-25. Que falha de circuito resultaria em V_{OUT} exatamente igual a V_A? Que falha faria com que V_{OUT} ficasse permanentemente em 0?

C, D

10-48. Utilize o CI CMOS 4016 (Seção 8-17) para implementar o chaveamento na Fig. 10-26, e projete a lógica de controle necessária para que cada entrada analógica seja convertida para seu equivalente digital em seqüência. O conversor A/D tem 10 bits e é do tipo de aproximações sucessivas utilizando um sinal de clock de 50 kHz, e requer um pulso de START com 10 μs de duração para iniciar cada conversão. As saídas digitais devem permanecer estáveis por 100 μs após a conversão terminar antes de chavear para o próximo valor analógico. Escolha uma freqüência de multiplexação adequada.

APLICAÇÃO EM MICROCOMPUTADOR

C, N, D

10-49. A Fig. 10-20 mostra como o conversor ADC0804 é interfaceado com um microcomputador. Ela mostra três sinais de controle, $\overline{CS}$, $\overline{RD}$ e $\overline{WR}$, que são gerados pelo microcomputador para o conversor A/D. Estes sinais são utilizados para iniciar cada nova conversão e para ler (transferir) os dados do A/D para o microcomputador através da barra de dados.

A Fig. 10-35 mostra um modo de implementar a lógica de decodificação de endereços. O sinal $\overline{CS}$ que ativa o ADC0804 é gerado pelas linhas de endereço de mais alta ordem do barramento de endereços do microprocessador. Sempre que o microprocessador desejar se comunicar com o ADC0804, ele coloca o endereço do ADC0804 na barra de endereços e a lógica de decodificação aciona o sinal $\overline{CS}$ em BAIXO. Note que, além das linhas de endereços, um sinal de temporização e controle (*ALE*) é conectado na entrada de habilitação $\overline{E_2}$. Sempre que *ALE* está em ALTO, significa que os endereços podem estar sofrendo uma transição de níveis, de modo que o decodificador deve permanecer desabilitado até que *ALE* vá para BAIXO (quando o endereço estará válido e estável). Isto serve apenas para temporização e não tem efeito sobre o endereçamento do conversor A/D.

(a) Determine o endereço do ADC0804.

(b) Modifique o circuito da Fig. 10-35 para colocar o ADC0804 no endereço hexadecimal E8XX.

(c) Modifique o circuito da Fig. 10-35 para colocar o ADC0804 no endereço hexadecimal FFXX.

D

10-50. Você tem disponível um conversor A/D de aproximações sucessivas de 10 bits (AD573), mas seu sistema necessita apenas de oito bits de resolução e você só tem uma porta de oito bits disponível no microprocessador. Você pode usar este conversor A/D, e, em caso afirmativo, quais das 10 linhas de dados você ligaria na porta?

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 10-1

1. Converte uma quantidade física não-elétrica em uma quantidade elétrica.

2. Converte uma tensão ou corrente analógica em uma representação digital.

3. Armazena-os; realiza cálculos e algumas outras operações com eles.

4. Converte os dados digitais em sua representação analógica.

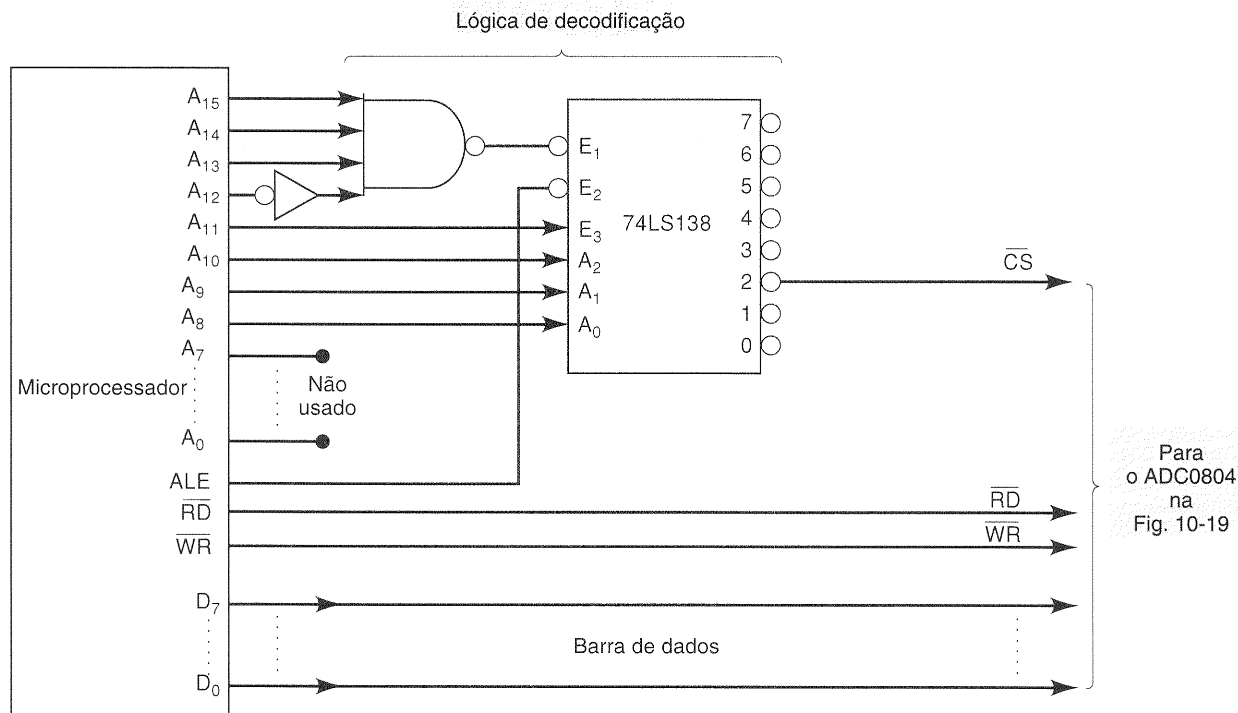


Fig. 10-35 Problema 10-49: interface entre um microprocessador e o ADC0804 da Fig. 10-19.

5. Controla uma variável física de acordo com um sinal elétrico de entrada.

SEÇÃO 10-2

1. 40 μA; 10,2 mA
2. 5,12 mA
3. 0,39%
4. 4.096
5. 12
6. 8,73 V
7. Verdadeiro
8. Produz um maior número de saídas analógicas possíveis entre 0 e o fundo de escala.

SEÇÃO 10-3

1. Utiliza apenas dois valores diferentes de resistores.
2. 640 kΩ
3. 0,5 V
4. Aumenta em 20%.

SEÇÃO 10-4

1. Desvio máximo do valor ideal de saída do conversor D/A, expresso como porcentagem do fundo de escala.
2. Tempo necessário para estabilizar a saída de um conversor D/A dentro de 1/2 degrau do seu valor de fundo de escala quando a entrada digital muda de 0 para o fundo de escala.
3. O erro de offset adiciona um pequeno valor positivo ou negativo constante na saída analógica de qualquer entrada digital.
4. Devido ao tempo de resposta do amplificador operacional do conversor corrente-tensão.

SEÇÃO 10-8

1. Informa a lógica de controle quando a saída do conversor D/A excede a entrada analógica.
2. Nas saídas do registrador.
3. Informa-nos quando a conversão termina e o equivalente digital de V_A está nas saídas do registrador.

SEÇÃO 10-9

1. A entrada digital para o conversor D/A é incrementada até que a saída em escada do D/A exceda a entrada analógica.
2. Um erro intrínseco causado pelo fato de V_{AX} não aumentar continuamente mas sim em degraus iguais à resolução do D/A. A tensão V_{AX} final pode ser diferente de V_A até por um degrau.
3. Se V_A aumenta, ele demora mais passos até que V_{AX} possa alcançar o primeiro degrau que excede V_A.
4. Verdadeiro
5. Circuito simples; tempo de conversão relativamente longo que varia com V_A.
6. $0010000111_2 = 135_{10}$ para ambos os casos

SEÇÃO 10-10

1. Processo de converter diferentes pontos de um sinal analógico para digital e armazenar os dados digitais para utilização posterior.
2. O computador gera o sinal de START para iniciar a conversão A/D do sinal analógico. Quando *EOC* vai para BAIXO, ele sinaliza o computador de que a conversão terminou. O computador armazena as saídas do conversor A/D na memória. O processo é repetido para o próximo ponto do sinal analógico.

SEÇÃO 10-11

1. O conversor A/D de aproximações sucessivas tem um tempo de conversão menor que não varia com V_A.
2. Tem uma lógica de controle mais complexa.
3. Falso
4. **(a)** 8
(b) 0-5 V
(c) $\overline{CS}$ controla o efeito dos sinais de $\overline{RD}$ e $\overline{WR}$; $\overline{WR}$ é utilizado para iniciar uma nova conversão; $\overline{RD}$ habilita os buffers de saída.
(d) Quando em BAIXO, sinaliza o fim da conversão.
(e) Separa o geralmente ruidoso terra digital do terra analógico para não contaminar o sinal analógico de entrada.
(f) Todas as tensões analógicas em $V_{in}(+)$ são medidas em relação a este pino. Permite que a faixa de entrada seja deslocada em relação a terra.

SEÇÃO 10-12

1. Verdadeiro
2. 4.095 comparadores e 4.096 resistores
3. A maior vantagem é sua velocidade de conversão; a desvantagem é o número necessário de componentes para uma resolução razoável.

SEÇÃO 10-13

1. Reduz o tempo de conversão utilizando um contador crescente/decrescente que permite que V_{AX} rastreie V_A sem iniciar do 0 para cada conversão.
2. VCO
3. Vantagens: baixo custo, imunidade a variações de temperatura. Desvantagem: tempo de conversão lento.
4. Conversor A/D flash, conversor A/D tensão-freqüência e de rampa dupla.
5. Um

SEÇÃO 10-14

1. Eles mantêm para o display os valores digitais apresentados pelos contadores no final da última conversão.
2. A subida de MONO1 transfere o resultado da conversão para os registradores; o pulso em MONO2 resseta os contadores para o início de um novo ciclo.
3. Tempo de conversão menor e constante.

SEÇÃO 10-15

1. Pega uma amostra do sinal de tensão analógico e armazena-o num capacitor.
2. Falso; eles são buffers de ganho unitário com alta impedância de entrada e baixa impedância de saída.

SEÇÃO 10-16

1. Utiliza apenas um conversor A/D.
2. Deveria ter módulo 8.

SEÇÃO 10-17

1. A forma de onda pode ser manipulada; pode mostrar partes da forma de onda ocorridas antes do disparo; tempo de armazenamento indefinido.
2. O conversor A/D digitaliza o ponto da forma de onda de entrada para armazenamento na memória; o conversor D/A vertical converte os dados armazenados de volta em tensões analógicas para produzir a deflexão vertical do feixe de elétrons; o conversor D/A horizontal produz uma tensão de varredura do tipo escada que provoca a deflexão horizontal do feixe de elétrons.

SEÇÃO 10-18

1. Filtragem de sinais analógicos.
2. Um conversor A/D
3. Para alterar sua resposta dinâmica, você simplesmente altera os números num programa de computador, e não tem que mexer em hardware.

CAPÍTULO 11

Dispositivos de Memória

■ SUMÁRIO

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Compreender e utilizar corretamente a terminologia associada com sistemas de memória.
- Descrever a diferença entre memórias de leitura/escrita e memórias somente de leitura.
- Discutir a diferença entre memórias voláteis e não-voláteis.
- Determinar a capacidade de um dispositivo de memória a partir de suas entradas e saídas.
- Resumir os passos que ocorrem quando a CPU lê ou escreve na memória.
- Distinguir entre os vários tipos de ROMs e citar algumas aplicações comuns.
- Compreender e descrever a organização e operação das RAMs estáticas e dinâmicas.
- Descrever a arquitetura dos tipos básicos de dispositivos de lógica programável e determinar o padrão de fusíveis necessário para a implementação de um dado conjunto de funções lógicas.
- Comparar as vantagens e desvantagens relativas às memórias EPROM, EEPROM e flash.
- Combinar CIs de memória para formar módulos de memória com maior capacidade e/ou tamanho de palavra.
- Utilizar os resultados de testes em sistemas com ROM ou RAM para determinar as possíveis falhas do sistema de memória.

■ INTRODUÇÃO

A principal vantagem dos sistemas digitais sobre os analógicos é a capacidade de armazenar, facilmente, grandes quantidades de informação e dados por períodos longos ou curtos. Esta capacidade de memória é o que torna os sistemas digitais tão versáteis e adaptáveis às diversas situações. Por exemplo, em um computador digital, a memória principal armazena instruções que informam ao computador o que fazer sob *qualquer* circunstância possível, de modo que o computador realizará sua tarefa com um mínimo de intervenção humana.

Este capítulo é dedicado ao estudo dos tipos mais comuns de dispositivos e sistemas de memória. Já estamos bem familiarizados com o flip-flop, que é um dispositivo eletrônico de memória. Também analisamos como grupos de FFs, chamados de *registradores*, podem ser utilizados para armazenar informação e como esta informação pode ser transferida para outros lugares. Registradores são elementos de memória de alto desempenho que são muito usados nas operações internas de um computador digital, no qual a informação digital está sendo continuamente transferida de um local para outro. Os avanços na tecnologia LSI e VLSI tornaram possível a obtenção de um grande número de FFs, em um único chip, organizados em vários arranjos de memória. Essas memórias semicondutoras bipolares e MOS são os dispositivos de memória mais rápidos disponíveis, e seus custos têm diminuído continuamente conforme a tecnologia LSI avança.

Dados digitais também podem ser armazenados como cargas em capacitores e um tipo de memória semicondutora muito importante utiliza esse princípio para obter altas densidades de armazenamento com requisitos de potência reduzidos.

Memórias semicondutoras são utilizadas como a **memória principal** de um computador (Fig. 11-1), na qual uma operação rápida é importante. A memória principal de um computador, também chamada de sua *memória de trabalho*, está em comunicação constante com a unidade central de processamento (CPU — *central processing unit*) à medida que um programa de instruções está sendo executado. Um programa e qualquer dado utilizado por ele permanecem na memória enquanto o computador está executando suas instruções. RAM e ROM (a serem definidas em breve) constituem a memória principal.

Uma outra forma de armazenamento no computador é realizada pela **memória auxiliar** (Fig. 11-1), que é separada da memória principal de trabalho. A memória auxiliar, também chamada de *armazenamento de massa*, tem a capacidade de armazenar uma enorme quantidade de dados sem a necessidade de energia elétrica. A memória auxiliar opera a uma velocidade bem mais baixa que a memória principal e armazena programas e dados que não estão sendo usados no momento pela CPU. Esta informação é transferida para a memória principal quando o computador precisa dela. Dispositivos comuns de memória auxiliar são o disco magnético, a fita magnética e o compact disk (CD). A memória semicondutora do tipo *flash*, com sua alta velocidade, baixos requisitos de potência, tamanho reduzido e operação que não envolve mecanismos, promete ser um importante competidor com as memórias de disco.

Vamos analisar detalhadamente as características dos dispositivos de memória mais comuns utilizados como memória interna de um computador. Primeiramente definiremos alguns termos comuns utilizados em sistemas de memória.

11-1 TERMINOLOGIA

O estudo dos sistemas e dos dispositivos de memória está repleto de termos que podem assustar um estudante. Antes de iniciarmos uma discussão pormenorizada sobre memórias, seria de grande ajuda que você compreendesse o significado de alguns termos mais básicos. Outros termos novos serão definidos conforme eles apareçam no capítulo.

- **Célula* de memória.** Um dispositivo ou circuito elétrico utilizado para armazenar um único bit (0 ou 1). Exemplos de célula de memória incluem um flip-flop, um

* Muitos autores definem célula de memória como sendo o conjunto de bits armazenado em uma determinada posição de memória, isto é, eles utilizam os conceitos de célula e palavra de memória de modo intercambiável. (*N.T.*)

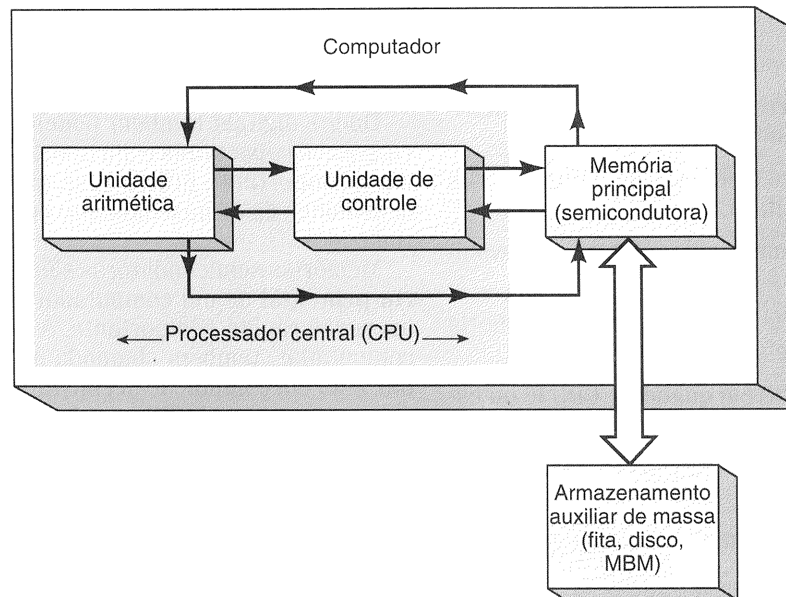


Fig. 11-1 Um computador normalmente utiliza memória principal de alta velocidade e memória auxiliar externa mais lenta.

capacitor carregado e um pequeno local numa fita ou disco magnético.

- **Palavra de memória.** Um grupo de bits (células) em uma memória que representa instruções ou dados de algum tipo. Por exemplo, um registrador de oito FFs pode ser considerado uma memória que está armazenando uma palavra de 8 bits. Os tamanhos de palavra nos computadores modernos variam tipicamente de 4 a 64 bits, dependendo do porte do computador.
- **Byte.** Um termo especial usado para um grupo de oito bits. Um byte sempre é constituído de 8 bits. Tamanhos de palavra podem ser expressos em bytes assim como em bits. Por exemplo, uma palavra de 8 bits é também uma palavra de um byte; uma palavra de 16 bits tem dois bytes, e assim por diante.
- **Capacidade.** Uma maneira de especificar quantos bits podem ser armazenados em um determinado dispositivo de memória ou num sistema de memória completo. Para ilustrar, suponha que temos uma memória capaz de armazenar 4.096 palavras de 20 bits. Isto representa uma capacidade total de 81.920 bits. Poderíamos também expressar essa capacidade de memória como 4.096 × 20. Quando representada desse modo, o primeiro número (4.096) é o número de palavras, e o segundo número (20) é o número de bits por palavra (tamanho da palavra). O número de palavras em uma memória freqüentemente é um múltiplo de 1.024. É comum usar a designação "1K" para representar $1.024 = 2^{10}$ quando nos referimos à capacidade de memória. Logo, uma memória com uma capacidade de armazenamento de 4K × 20 é na verdade uma memória de 4.096 × 20. O desenvolvimento de memórias maiores trouxe a designação "1M" ou "1 mega" para representar $2^{20} = 1.048.576$. Assim, uma memória que possui uma capacidade de 2M × 8 tem na verdade uma capacidade de 2.097.152 × 8. A designação "giga" se refere a $2^{30} = 1.073.741.824$.

EXEMPLO 11-1A

Um certo chip de memória semicondutora é especificado como 2K × 8. Quantas palavras podem ser armazenadas neste chip? Qual é o tamanho da palavra? Quantos bits este chip pode armazenar no total?

Solução

$$2K = 2 \times 1.024 = 2.048 \text{ palavras}$$

Cada palavra tem 8 bits (um byte). O número total de bits, portanto, é

$$2.048 \times 8 = 16.384 \text{ bits}$$

EXEMPLO 11-1B

Qual das memórias armazena mais bits: uma memória de 5M × 8 ou uma memória que armazena 1M palavras com um tamanho de palavra de 16 bits?

Solução

$$5M \times 8 = 5 \times 1.048.576 \times 8 = 41.943.040 \text{ bits}$$
$$1M \times 16 = 1.048.576 \times 16 = 16.777.216 \text{ bits}$$

A memória de 5M × 8 armazena mais bits.

- **Densidade.** Um outro termo para *capacidade*. Quando dizemos que um dispositivo de memória tem uma densidade maior do que um outro, queremos dizer que ele pode armazenar mais bits no mesmo espaço. Ele é mais denso.
- **Endereço.** Um número que identifica a posição de uma palavra na memória. Cada palavra armazenada em um

Endereços	
000	Palavra 0
001	Palavra 1
010	Palavra 2
011	Palavra 3
100	Palavra 4
101	Palavra 5
110	Palavra 6
111	Palavra 7

Fig. 11-2 Cada posição tem um endereço binário específico.

dispositivo ou sistema de memória possui um endereço único. Endereços sempre existem num sistema digital como um número binário, embora, por conveniência, números em octal, hexadecimal e decimal sejam freqüentemente utilizados para representar o endereço. A Fig. 11-2 ilustra uma pequena memória constituída de oito palavras. Cada uma destas oito palavras tem um endereço específico representado por um número de três bits que varia de 000 até 111. Sempre que nos referimos a uma posição específica na memória, utilizamos seu código de endereço para identificá-la.

- **Operação de Leitura.** Operação na qual a palavra binária armazenada numa determinada posição (endereço) de memória é detectada e então transferida para outro dispositivo. Por exemplo, se desejamos utilizar a palavra 4 da memória da Fig. 11-2 para algum propósito, devemos realizar uma operação de leitura no endereço 100. A operação de leitura freqüentemente é chamada de operação de *busca*, pois a palavra está sendo buscada da memória. Utilizaremos os dois termos indistintamente.
- **Operação de Escrita.** Operação na qual uma nova palavra é colocada numa determinada posição de memória. Também é chamada de operação de *armazenamento*. Sempre que uma nova palavra é escrita numa posição de memória, ela substitui a palavra que estava previamente armazenada lá.
- **Tempo de Acesso.*** Uma medida da velocidade de operação de um dispositivo de memória. É o tempo necessário para realizar uma operação de leitura. Mais especificamente, é o tempo entre a memória receber uma nova entrada de endereço e os dados se tornarem disponíveis na saída da memória. O símbolo t_{ACC} é utilizado para tempo de acesso.
- **Memória Volátil.** Qualquer tipo de memória que necessita da aplicação de energia para poder armazenar informação. Se a energia elétrica é removida, todas as informações armazenadas na memória são perdidas. Muitas memórias semicondutoras são voláteis, enquanto todas as memórias magnéticas são *não-voláteis*, o que significa que elas podem armazenar informação sem energia elétrica.
- **Memória de Acesso Aleatório (RAM — *Random-Access Memory*).** Memória na qual a posição física real de uma palavra da memória não tem efeito sobre o tempo necessário para ler ou escrever nesta posição. Em outras palavras, o tempo de acesso é o mesmo para qualquer endereço na memória. A maioria das memórias semicondutoras é de acesso aleatório.
- **Memória de Acesso Seqüencial (SAM — *Sequential-Access Memory*).** Um tipo de memória no qual o tempo de acesso não é constante mas varia dependendo do endereço. Uma determinada palavra armazenada é encontrada percorrendo todos os endereços até que o endereço desejado seja alcançado. Isto produz tempos de acesso que são muito maiores do que os das memórias de acesso aleatório. Um exemplo de dispositivo de memória de acesso seqüencial é uma fita magnética. Para ilustrar a diferença entre SAM e RAM, considere a situação na qual você gravou 60 minutos de música numa fita cassete de áudio. Quando desejar alcançar uma música em particular, você terá que retroceder ou avançar a fita até a encontrar. O processo é relativamente lento, e o tempo necessário depende de onde a música desejada está gravada na fita. Isto é SAM, já que você percorreu através das informações registradas até encontrar o que estava procurando. A contrapartida RAM para isso seria um CD de áudio, no qual você pode rapidamente selecionar qualquer música informando o código apropriado, e ele gasta aproximadamente o mesmo tempo, não importando a música selecionada. As memórias de acesso seqüencial são utilizadas onde os dados a serem acessados sempre vêm numa longa seqüência de palavras sucessivas. A memória de vídeo, por exemplo, deve fornecer seu conteúdo na mesma ordem repetidamente para manter a imagem na tela.
- **Memória de Leitura e Escrita (RWM — *Read/Write Memory*).** Qualquer memória que possa ser lida ou escrita de maneira igualmente fácil.
- **Memória Somente de Leitura (ROM — *Read-Only Memory*).** Uma vasta classe de memórias semicondutoras, projetadas para aplicações nas quais a razão entre as operações de leitura e escrita é muito alta. Tecnicamente, uma ROM pode ser escrita (programada) apenas uma vez, e esta operação normalmente é realizada na fábrica. Depois disso, as informações podem ser somente lidas da memória. Outros tipos de ROM são na verdade RMM (*read-mostly memories*), nas quais se pode escrever mais de uma vez; porém a operação de escrita é mais complicada do que a de leitura, e não é realizada freqüentemente. Os vários tipos de ROM serão discutidos posteriormente. *Todas as ROMs são não-voláteis* e armazenam dados quando a energia é removida.
- **Dispositivos de Memória Estática.** Dispositivos de memória semicondutora nos quais os dados permanecem armazenados enquanto a energia está presente, sem a necessidade de reescrever periodicamente os dados na memória.
- **Dispositivos de Memória Dinâmica.** Dispositivos de memória semicondutora nos quais os dados *não* permanecem armazenados, mesmo com a energia presente, a menos que os dados sejam periodicamente reescritos na memória. Esta última operação é denominada *refresh*.
- **Memória Principal.** Também chamada de *memória de trabalho* do computador. Ela armazena instruções e dados que a CPU está acessando no momento. É a memó-

* O mais comum é a utilização do termo tempo de acesso tanto para escrita quanto para leitura, isto é, o tempo entre a memória receber um novo endereço e a realização da operação (leitura ou escrita). (*N.T.*)

ria mais rápida num computador e sempre é uma memória semicondutora.

- **Memória Auxiliar.** Também chamada de *memória de massa* porque ela armazena grandes quantidades de informação externamente à memória principal. É mais lenta do que a memória principal e sempre é não-volátil. Discos magnéticos e CDs são dispositivos comuns de memória auxiliar.

Questões de Revisão

1. Defina os seguintes termos:
   (a) *Célula de memória*
   (b) *Palavra de memória*
   (c) *Endereço*
   (d) *Byte*
   (e) *Tempo de acesso*
2. Uma certa memória tem capacidade de 8K × 16. Quantos bits ela tem em cada palavra? Quantas palavras estão sendo armazenadas? Quantas células esta memória contém?
3. Explique a diferença entre as operações de leitura e escrita.
4. *Verdadeiro ou falso*: Uma memória volátil perderá seus dados armazenados quando a energia elétrica for interrompida.
5. Explique a diferença entre SAM e RAM.
6. Explique a diferença entre RWM e ROM.
7. *Verdadeiro ou falso*: Uma memória dinâmica manterá seus dados enquanto a energia elétrica estiver presente.

11-2 PRINCÍPIOS DE OPERAÇÃO DA MEMÓRIA

Embora cada tipo de memória seja diferente na sua operação interna, certos princípios básicos de operação são iguais para todos os sistemas de memória. Uma compreensão dessas idéias básicas ajudará nosso estudo sobre os dispositivos de memória.

Todos os sistemas de memória necessitam de diversos tipos diferentes de linhas de entrada e de saída para realizar as seguintes funções:

1. Selecionar o endereço na memória que está sendo acessado para uma operação de leitura ou escrita.
2. Selecionar uma operação de leitura ou uma de escrita a ser realizada.
3. Fornecer os dados de entrada a serem armazenados na memória durante uma operação de escrita.
4. Manter os dados de saída vindos da memória durante uma operação de leitura.
5. Habilitar (ou desabilitar) a memória de modo que ela responda (ou não) às entradas de endereçamento e ao comando de leitura/escrita.

A Fig. 11-3(a) ilustra essas funções básicas num diagrama simplificado de uma memória de 32 × 4 que armazena trinta e duas palavras de quatro bits. Tendo em vista que o tamanho da palavra é de quatro bits, existem quatro linhas de entrada de dados, I_0 a I_3, e quatro linhas de saída de dados, O_0 a O_3. Durante uma operação de escrita, o dado a ser armazenado na memória deve ser aplicado nas linhas de entrada de dados. Durante uma operação de leitura, a palavra que está sendo lida da memória aparece nas linhas de saída de dados.

Entradas de Endereço

Como esta memória armazena 32 palavras, ela tem 32 posições de armazenamento diferentes, e portanto possui 32 endereços binários diferentes, variando de 00000 até 11111 (0 a 31 em decimal). Logo, existem cinco entradas de endereço, A_0 até A_4. Para acessar uma das posições de memória para uma operação de leitura ou escrita, o código de endereçamento de cinco bits para essa posição é aplicado nas entradas de endereço. De um modo geral, N entradas de endereço são necessárias para uma memória que possui uma capacidade de 2^N palavras.

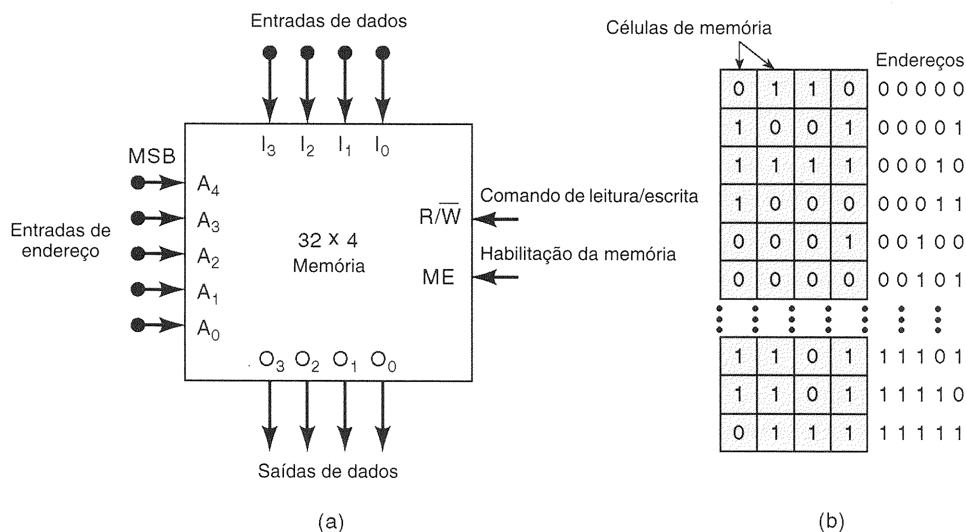


Fig. 11-3 (a) Diagrama de uma memória de 32 × 4; (b) arranjo virtual das células de memória em 32 palavras de quatro bits.

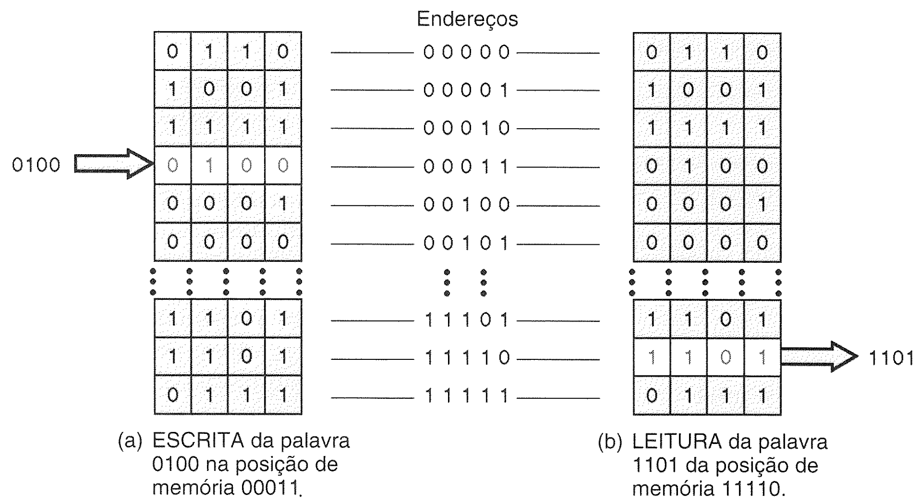


Fig. 11-4 Ilustração simplificada das operações de leitura e escrita numa memória de 32 × 4: (a) escrita da palavra 0100 na posição de memória 00011; (b) leitura da palavra 1101 da posição de memória 11110.

Podemos visualizar a memória da Fig. 11-3(a) como um arranjo de 32 registradores, no qual cada registrador guarda uma palavra de quatro bits, conforme mostra a Fig. 11-3(b). Cada posição é mostrada contendo quatro células de memória, que guardam 1s ou 0s, que formam a palavra de dados armazenada nesta posição. Por exemplo, a palavra de dados 0110 está armazenada no endereço 00000, a palavra de dados 1001 está armazenada no endereço 00001, e assim por diante.

A Entrada $R/\overline{W}$

Esta entrada controla qual operação deve ser realizada na memória: leitura (R — *Read*) ou Escrita (W — *Write*). A entrada é identificada por $R/\overline{W}$, e, como não existe a barra sobre *R*, isto indica que a operação de leitura ocorre quando $R/\overline{W} = 1$. A barra sobre *W* indica que a operação de escrita acontece quando $R/\overline{W} = 0$. Outros identificadores são usados freqüentemente para essa entrada. Dois dos mais comuns são $\overline{W}$ (escrita) e $\overline{WE}$ *(write enable* — habilitação de escrita). Novamente, a barra indica que a operação de escrita ocorre quando a entrada está em BAIXO. Fica subentendido que a operação de leitura ocorre para nível ALTO.

Uma ilustração simplificada das operações de leitura e escrita é mostrada na Fig. 11-4. A parte (a) mostra o dado 0100 sendo escrito no registrador de memória da posição de endereço 00011. Este dado deve ser aplicado nas linhas de entrada de dados, e substitui o dado previamente armazenado no endereço 00011. A parte (b) mostra a palavra 1101 sendo lida do endereço 11110. Esta palavra deve aparecer nas linhas de saída de dados da memória. Após a operação de leitura, a palavra 1101 ainda está armazenada no endereço 11110. Isto é, a operação de leitura não altera o dado armazenado.

Habilitação da Memória

Muitos sistemas de memória têm algum modo de desabilitar completamente uma parte ou toda a memória, de modo que ela não responde às outras entradas. Isto é representado na Fig. 11-3 pela entrada *ME*, embora ela possa ter nomes diferentes nos vários sistemas de memória, tais como *chip enable (CE)* ou *chip select (CS)*. Aqui ela é mostrada como uma entrada ativa em ALTO que habilita a memória a operar normalmente quando é mantida em ALTO. Um nível BAIXO nesta entrada desabilita a memória, de modo que ela não responderá às entradas de endereço e de $R/\overline{W}$. Esse tipo de entrada é útil quando vários módulos de memória são combinados para formar uma memória maior. Examinaremos essa idéia mais tarde.

EXEMPLO 11-2

Descreva as condições de cada entrada e saída quando o conteúdo da posição cujo endereço é 00100 deve ser lido.

Solução

Entradas de endereço: 00100
Entradas de dados: xxxx (não utilizadas)
$R/\overline{W}$: nível ALTO
HABILITAÇÃO DE MEMÓRIA (ME): ALTO
Saídas de dados: 0001

EXEMPLO 11-3

Descreva as condições de cada entrada e saída quando a palavra 1110 deve ser escrita na posição de endereço 01101.

Solução

Entradas de endereço: 01101
Entradas de dados: 1110
$R/\overline{W}$: BAIXO

HABILITAÇÃO DE MEMÓRIA (*ME*): ALTO
Saídas de dados: xxxx (não utilizadas; usualmente em alta impedância)

EXEMPLO 11-4

Uma determinada memória tem uma capacidade de 4K × 8.

(a) Quantas linhas de entrada de dados e saída de dados ela tem?
(b) Quantas linhas de endereço ela tem?
(c) Qual é a sua capacidade em bytes?

Solução

(a) Oito de cada, pois o tamanho da palavra é oito.
(b) A memória armazena 4K = 4 × 1.024 = 4.096 palavras. Assim, existem 4.096 endereços de memória. Tendo em vista que 4.096 = 2^{12}, ela necessita de um código de endereço de 12 bits para especificar um entre 4.096 endereços.
(c) Um byte tem oito bits. Esta memória tem uma capacidade de 4.096 bytes.

A memória do exemplo da Fig. 11-3 ilustra as importantes funções de entrada e saída comuns na maioria dos sistemas de memória. Naturalmente, cada tipo de memória pode ter outras linhas de entrada ou saída peculiares. Isto será discutido à medida que apresentarmos os tipos específicos de memórias.

Questões de Revisão

1. Quantas entradas de endereço, entradas de dados e saídas de dados são necessárias para uma memória de 16K × 12?
2. Qual é a função da entrada $R/\overline{W}$?
3. Qual é a função da entrada de HABILITAÇÃO DE MEMÓRIA?

11-3 CONEXÕES CPU—MEMÓRIA

A parte principal deste capítulo é dedicada a memória semicondutora, que, conforme dito anteriormente, implementa a memória principal dos computadores mais modernos. Lembre-se de que esta memória principal mantém comunicação constante com a CPU (*central processing unit*). Não é necessário estar familiarizado com a operação detalhada de uma CPU agora, e, portanto, o tratamento simplificado, apresentado a seguir, da interface entre CPU e memória fornecerá a base necessária para facilitar o entendimento do nosso estudo de dispositivos de memória.

A memória principal de um computador é constituída de CIs de RAM e ROM que são interfaceados com a CPU através de três grupos de linhas de sinais ou barramentos. Isto é mostrado na Fig. 11-5 como as linhas de endereço ou barramento de endereço, as linhas de dados ou barramento de dados e as linhas de controle ou barramento de controle. Cada um destes barramentos consiste em várias linhas (note que são representados por uma única linha com uma barra), e o número de linhas em cada barramento varia de um computador para outro. Os três barramentos são fundamentais para permitir que a CPU escreva dados na memória ou leia dados da memória.

Quando um computador está executando um programa, a CPU continuamente busca (lê) informação dessas posições de memória que contêm (1) os códigos do programa que representam as operações a serem realizadas e (2) os dados para serem manipulados. A CPU também armazenará (escreverá) dados nas posições de memória conforme indicado pelas instruções do programa. Sempre que a CPU deseja escrever dados em uma posição de memória, os seguintes passos devem ocorrer:

Operação de Escrita

1. A CPU fornece o endereço binário da posição de memória na qual o dado deve ser armazenado. Ela coloca este endereço nas linhas do barramento de endereço.
2. A CPU coloca o dado a ser armazenado nas linhas do barramento de dados.
3. A CPU ativa as linhas de controle apropriadas para a operação de escrita na memória.
4. Os CIs de memória decodificam o endereço binário para determinar qual posição está sendo selecionada para a operação de armazenamento.

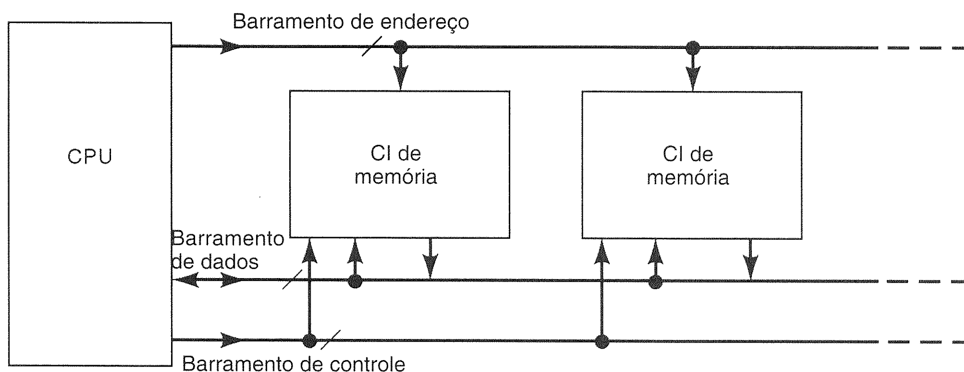


Fig. 11-5 Três grupos de linhas (barramentos) conectam os CIs da memória principal na CPU.

5. Os dados no barramento de dados é transferido para a posição de memória selecionada.

Sempre que a CPU deseja ler um dado de uma posição de memória, os seguintes passos devem ocorrer:

Operação de Leitura
1. A CPU fornece o endereço binário da posição de memória da qual o dado deve ser recuperado. Ela coloca este endereço nas linhas do barramento de endereço.
2. A CPU ativa as linhas de controle apropriadas para a operação de leitura da memória.
3. Os CIs de memória decodificam o endereço binário para determinar qual posição está sendo selecionada para a operação de leitura.
4. Os CIs de memória colocam os dados da posição de memória selecionada no barramento de dados, de onde são transferidos para a CPU.

Esses passos devem ter esclarecido a função de cada um dos barramentos do sistema:

- **Barramento de endereço.** Este é um barramento *unidirecional* que leva as saídas binárias de endereço da CPU para os CIs de memória para selecionar uma posição de memória.
- **Barramento de dados.** Este é um barramento *bidirecional* que transporta dados entre a CPU e os CIs de memória.
- **Barramento de controle.** Este barramento leva sinais de controle (tais como o sinal $R/\overline{W}$) da CPU para os CIs de memória.

À medida que apresentarmos os CIs de memória reais, examinaremos a atividade dos sinais que aparecem nesses barramentos para as operações de leitura e escrita.

Questões de Revisão

1. Cite o nome dos três grupos de linhas que conectam a CPU com a memória interna.
2. Relacione os passos que ocorrem quando a CPU lê da memória.
3. Relacione os passos que ocorrem quando a CPU escreve na memória.

11-4 MEMÓRIAS SOMENTE DE LEITURA

A memória somente de leitura é um tipo de memória semicondutora que é projetada para armazenar dados imutáveis ou aqueles que não mudam com freqüência. Durante a operação normal, novos dados não podem ser escritos na ROM; somente os dados já armazenados podem ser lidos. Para alguns tipos de ROMs, os dados que estão armazenados devem ser colocados na memória durante o processo de fabricação; para outras ROMs, os dados podem ser colocados eletricamente. O processo de colocar os dados é chamado de **programação** ou *"queima"* da ROM. Algumas ROMs não podem ter seus dados modificados uma vez programados. Outras podem ser *apagadas* e reprogramadas tantas vezes quanto necessário. Estudaremos os vários tipos de ROMs, em detalhe, mais adiante. Por enquanto, vamos considerar que as ROMs já foram programadas e que estão mantendo os dados armazenados.

As ROMs são usadas para armazenar dados e informações que não devem ser alterados durante a operação normal de um sistema. A principal aplicação das ROMs é no armazenamento de programas em microcomputadores. Uma vez que todas as ROMs são *não-voláteis*, estes programas não são perdidos quando a fonte de alimentação é desligada. Quando o microcomputador é ligado, ele pode imediatamente começar a execução do programa armazenado na ROM. ROMs também são usadas em equipamentos controlados por microprocessador, tais como: caixas registradoras eletrônicas, aparelhos domésticos e sistemas de segurança.

Diagrama de Blocos de uma ROM

Um diagrama de blocos típico para uma ROM pode ser visto na Fig. 11-6(a). Ele possui três grupos de sinais: entradas de endereço, entradas de controle e entradas de dados. Com base no que foi dito anteriormente, podemos dizer que esta ROM está armazenando 16 palavras, uma vez que $2^4 = 16$ endereços possíveis, e que cada palavra contém 8 bits, pois existem oito saídas de dados. Portanto, esta é uma ROM de 16×8. Uma outra maneira de descrever a capacidade da ROM é dizer que ela armazena 16 bytes de dados.

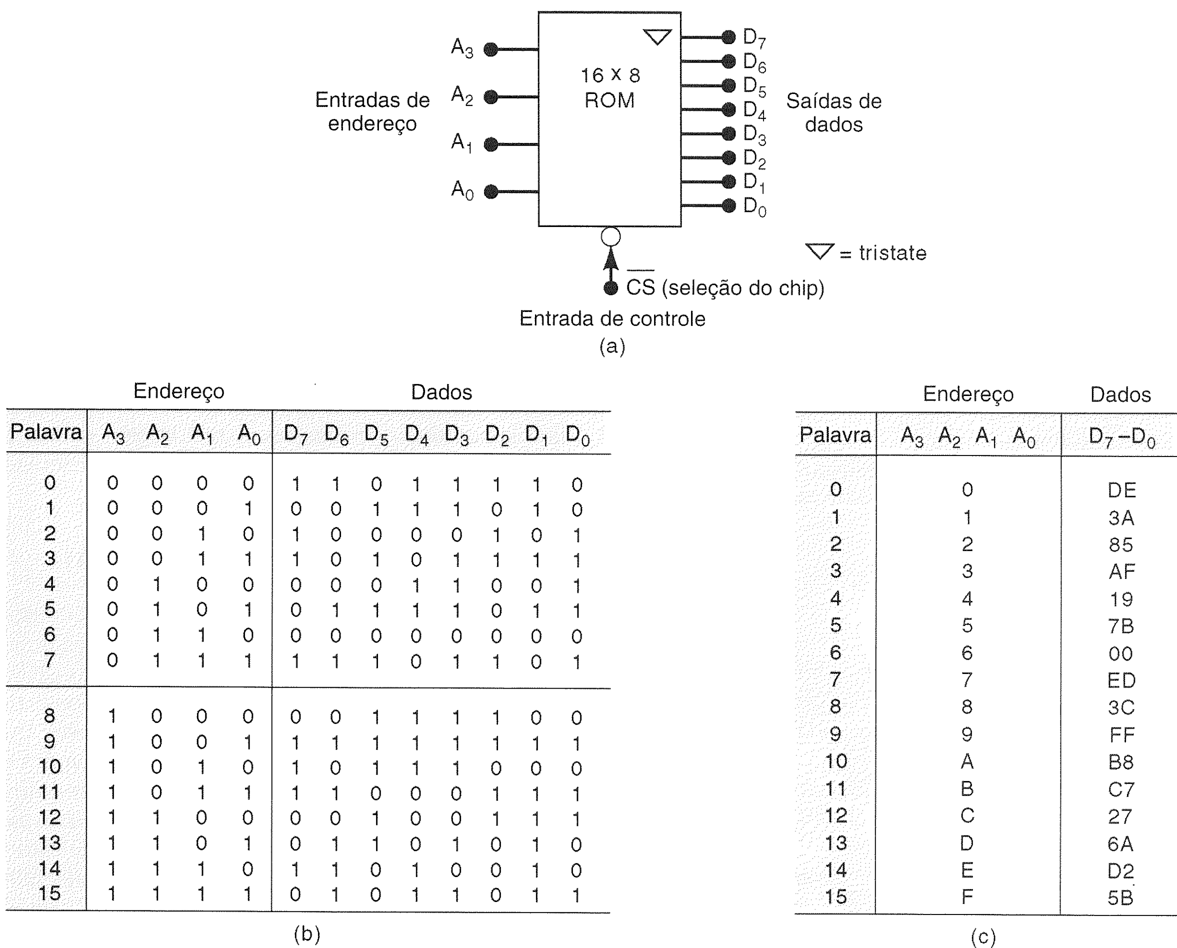
As saídas de dados mais comuns da maioria dos CIs de ROMs são do tipo tristate, para permitir a conexão de vários chips de ROMs num mesmo barramento de dados para expansão de memória. Os números de saídas de dados mais comuns são 4, 8 e 16 bits, sendo que palavras de 8 bits são as mais comuns.

A entrada de controle $\overline{CS}$ (*chip select*) é a entrada de **seleção do chip**. Ela é uma entrada que habilita ou não as saídas da ROM. Alguns fabricantes usam outros nomes para esta entrada de controle, tais como: *CE* (*chip enable* — habilitação do chip) ou *OE* (*output enable* — habilitação da saída). Muitas ROMs têm duas ou mais entradas de controle que devem estar ativas para que as saídas de dados sejam habilitadas e o conteúdo do endereço selecionado possa ser lido. Em alguns CIs de ROMs, uma das entradas de controle, geralmente *CE*, é utilizada para colocar a ROM em modo *standby* quando ela não estiver sendo usada. Isto diminui o fornecimento de corrente para a memória a partir da fonte de alimentação do sistema.

A entrada $\overline{CS}$, mostrada na Fig. 11-16(a), é ativa em nível BAIXO, e portanto deve estar em nível BAIXO para permitir que os dados da ROM apareçam nas saídas. Observe que não há entrada $R/\overline{W}$ (leitura/escrita), porque não se pode escrever em uma ROM durante a operação normal.

A Operação de Leitura

Vamos considerar que uma ROM foi programada com os dados mostrados na tabela da Fig. 11-6(b). As dezesseis palavras de dados são armazenadas em 16 endereços diferentes. Por exemplo, a palavra de dados armazenada na posição 0011 é 10101111. É claro, os dados estão armaze-



	Endereço				Dados							
Palavra	A_3	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	1	1	0	1	1	1	1	0
1	0	0	0	1	0	0	1	1	1	0	1	0
2	0	0	1	0	1	0	0	0	0	1	0	1
3	0	0	1	1	1	0	1	0	1	1	1	1
4	0	1	0	0	0	0	0	1	1	0	0	1
5	0	1	0	1	0	1	1	1	1	0	1	1
6	0	1	1	0	0	0	0	0	0	0	0	0
7	0	1	1	1	1	1	1	0	1	1	0	1
8	1	0	0	0	0	0	1	1	1	1	0	0
9	1	0	0	1	1	1	1	1	1	1	1	1
10	1	0	1	0	1	0	1	1	1	0	0	0
11	1	0	1	1	1	1	0	0	0	1	1	1
12	1	1	0	0	0	0	1	0	0	1	1	1
13	1	1	0	1	0	1	1	0	1	0	1	0
14	1	1	1	0	1	1	0	1	0	0	1	0
15	1	1	1	1	0	1	0	1	1	0	1	1

(b)

	Endereço	Dados
Palavra	A_3 A_2 A_1 A_0	D_7-D_0
0	0	DE
1	1	3A
2	2	85
3	3	AF
4	4	19
5	5	7B
6	6	00
7	7	ED
8	8	3C
9	9	FF
10	A	B8
11	B	C7
12	C	27
13	D	6A
14	E	D2
15	F	5B

(c)

Fig. 11-6 (a) Símbolo de uma ROM típica; (b) tabela mostrando os dados binários para cada endereço; (c) a mesma tabela em hexadecimal.

nados de modo binário no interior da ROM, mas freqüentemente usaremos a notação hexadecimal para mostrar de modo compacto os dados programados. Isto é feito na Fig. 11-6(c).

Para ler uma palavra de dados armazenados na ROM, necessitamos: (1) fornecer valores apropriados nas entradas de endereço e depois (2) ativar as entradas de controle. Por exemplo, se desejamos ler os dados armazenados na posição 0111 da ROM da Fig. 11-6, devemos aplicar $A_3A_2A_1A_0$ = 0111 nas entradas de endereço e depois colocar $\overline{CS}$ em BAIXO.

As entradas de endereço serão decodificadas no interior da ROM para selecionar a palavra de dados correta, isto é, 11101101, que irá aparecer nas saídas D_7 a D_0. Se a entrada $\overline{CS}$ for mantida em ALTO, as saídas da ROM não estarão habilitadas e ficarão em alta impedância.

Questões de Revisão

1. *Verdadeiro ou falso*: Todas as ROMs são não-voláteis.
2. Descreva o procedimento de leitura de uma ROM.
3. O que é *programação* ou *queima* de uma ROM?

11-5 ARQUITETURA DA ROM

A arquitetura (estrutura) interna de um circuito integrado de uma ROM é bastante complexa. Além disso, não precisaremos nos familiarizar com todos os seus detalhes. Entretanto, é importante estudar um diagrama simplificado como o que está na Fig. 11-7 para uma ROM de 16 × 8.

Matriz de Registradores

A matriz de registradores armazena os dados que foram programados na ROM. Cada registrador contém um número de células de memória igual ao tamanho da palavra utilizada. Neste caso, cada registrador armazena uma palavra de 8 bits. Os registradores estão organizados como uma matriz quadrada que é comum em muitos chips de memória. Podemos especificar a posição de cada registrador como se ele estivesse em uma linha e em uma coluna específica. Por exemplo, o registrador 0 está na linha 0, coluna 0, e o registrador 9 está na linha 1, coluna 2.

Os oitos bits de dados de saída de cada registrador são conectados a um barramento interno de dados que percorre todo o circuito. Cada registrador tem duas entradas de habilitação (E); ambas devem estar em ALTO para que os

dados do registrador sejam colocados no barramento de dados.

Decodificadores de Endereço

O endereço $A_3A_2A_1A_0$ determina que registrador na matriz será habilitado para colocar seus dados no barramento. Os bits de endereço, A_1A_0, são fornecidos como entradas a um decodificador 2 para 4 que ativa uma linha de seleção de coluna. Apenas um registrador terá a sua linha e a sua coluna selecionadas pelas entradas de endereço, e então estará habilitado.

EXEMPLO 11-5

Qual o registrador que será habilitado pelo endereço 1101?

Solução

A_3A_2 = 11 fazem com que o decodificador de colunas ative a linha de seleção da coluna 3, e A_1A_0 = 01 fazem com que o decodificador de linhas ative a linha de seleção da linha 1. Isto coloca níveis ALTOs em ambas as entradas de habilitação do registrador 13, permitindo que seu conteúdo seja colocado no barramento. Observe que os outros registradores da coluna 3 terão apenas uma entrada de habilitação ativa. O mesmo é válido para os outros registradores da linha 1.

EXEMPLO 11-6

Que endereço irá habilitar o registrador 7?

Solução

As entradas de habilitação desse registrador estão conectadas nos sinais de seleção da linha 3 e da coluna 1, respectivamente. Para selecionar a linha 3, as entradas A_1A_0 devem estar em 11, e para selecionar a coluna 1, as entradas A_3A_2 devem estar em 01. Então, o endereço necessário será $A_3A_2A_1A_0$ = 0111.

Buffers de Saída

O registrador que está habilitado pelas entradas de endereço coloca seu conteúdo no barramento de dados. Estes dados vão para os buffers de saída, que colocarão os dados na

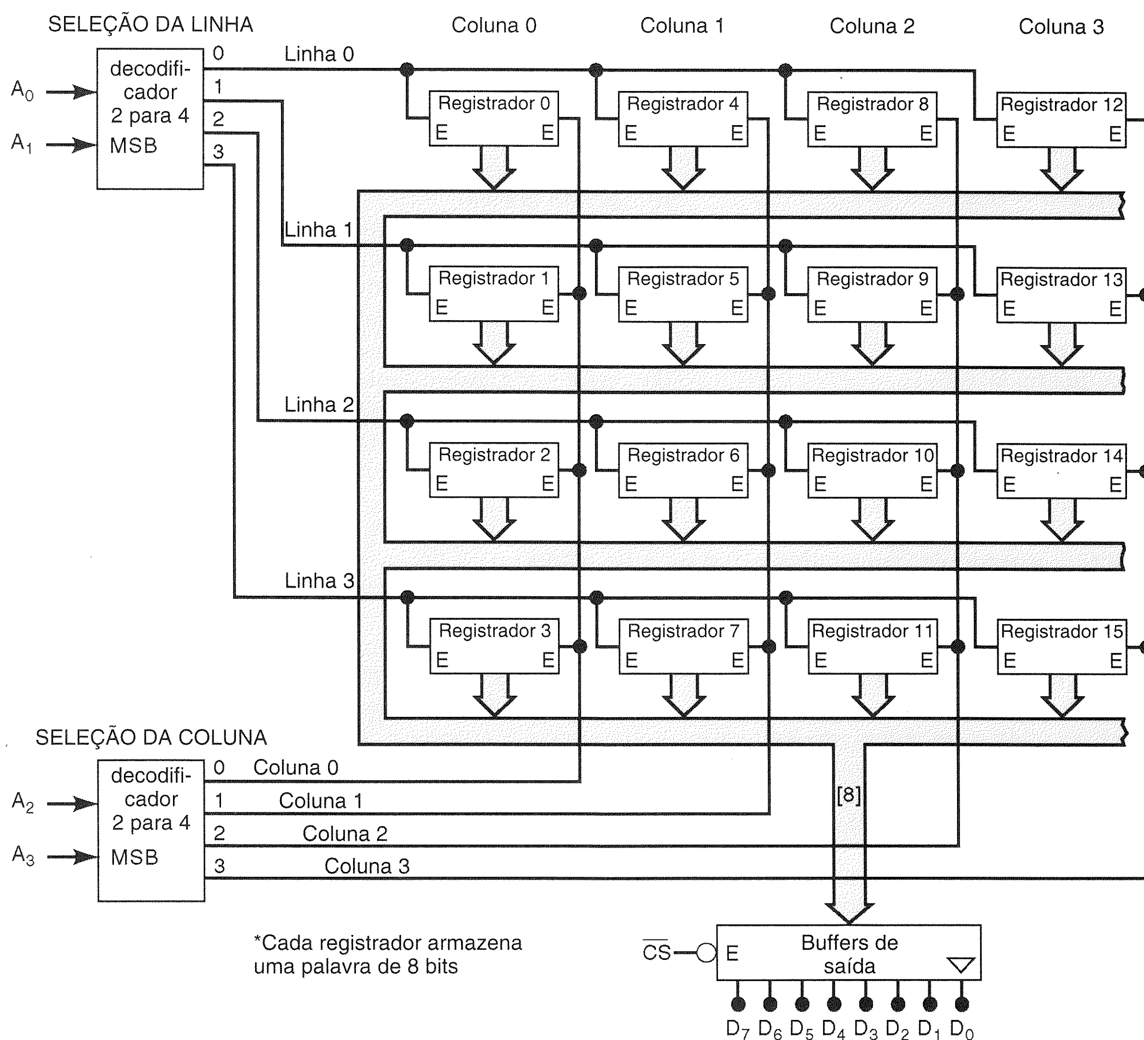


Fig. 11-7 Estrutura de uma ROM de 16 × 8.

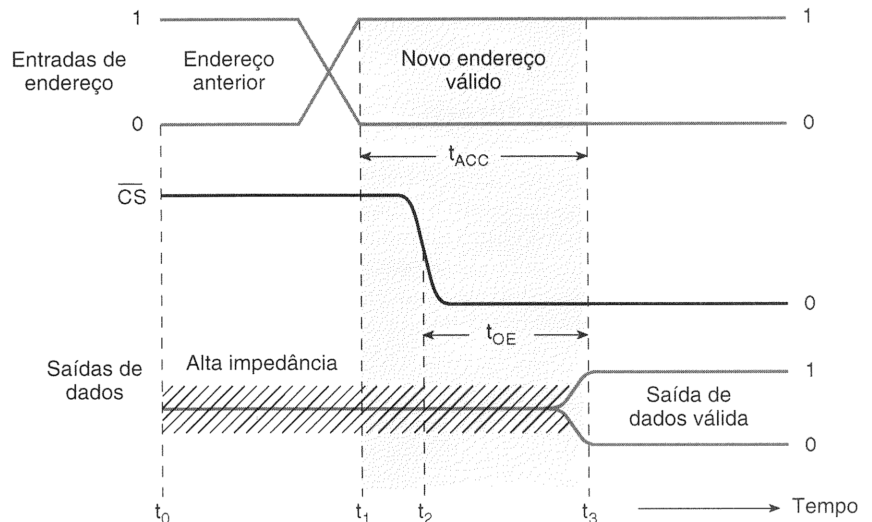


Fig. 11-8 Temporização típica para uma operação de leitura de uma ROM.

saída de dados externa, desde que $\overline{CS}$ esteja em BAIXO. Se $\overline{CS}$ estiver em ALTO, os buffers de saída estarão em alta impedância, e D_7 a D_0 ficarão em flutuação.

A arquitetura mostrada na Fig. 11-7 é semelhante àquela encontrada em muitos CIs de ROMs. Dependendo do número de palavras armazenadas, os registradores em algumas ROMs não são organizados na forma de uma matriz quadrada. Por exemplo, a ROM MOS 2764 da Intel armazena 8.192 palavras de oito bits. Seus 8.192 registradores são organizados em uma matriz de 256 linhas de 32 registradores cada. A capacidade de armazenamento de uma ROM se situa em uma faixa entre 256 × 4 até 8M × 8.

EXEMPLO 11-7

Descreva a arquitetura interna de uma ROM que armazena 4Kbytes em uma matriz quadrada.

Solução

4K é na verdade 4 × 1.024 = 4.096, portanto essa ROM armazena 4.096 palavras de oito bits. Podemos dizer que cada uma das palavras está armazenada em um registrador de oito bits, logo existem 4.096 registradores conectados a um barramento de dados interno ao chip. Uma vez que 4.096 = 64^2, os registradores estão organizados em uma matriz 64 × 64, isto é, uma matriz de 64 linhas e 64 colunas. Isto requer um decodificador de 6 para 64 para decodificar as seis entradas de endereço para selecionar a linha e um segundo decodificador 6 para 64 para decodificar as outras seis entradas de endereço que selecionam a coluna. Então, um total de 12 entradas de endereço é necessário. Isto faz sentido, uma vez que 2^{12} = 4.096, e existem 4.096 endereços diferentes.

Questões de Revisão

1. Qual o endereço que é necessário fornecer para ler os dados do registrador 9 na Fig. 11-7?
2. Descreva a função do decodificador de seleção de linha, do decodificador de seleção de coluna e dos buffers de saída na arquitetura da ROM.

11-6 TEMPORIZAÇÃO DA ROM

Existe um atraso de propagação entre a aplicação dos sinais de entrada e o aparecimento dos dados na saída, durante uma operação de leitura. Este atraso, chamado de **tempo de acesso**, t_{ACC}, é uma medida da velocidade de operação da ROM. O tempo de acesso é representado graficamente pelas formas de onda da Fig. 11-8.

A forma de onda superior representa as entradas de endereço; a do meio é o sinal de seleção do chip, $\overline{CS}$, que é ativo em BAIXO; e a forma de onda inferior representa os dados de saída. No instante t_0, as entradas de endereço estão todas em algum nível específico, algumas em ALTO, algumas em BAIXO. $\overline{CS}$ está em ALTO, de modo que as saídas estão em alta impedância (representada pela linha hachurada).

Imediatamente antes de t_1, as entradas de endereço estão mudando para um novo valor para realizar uma nova leitura. No instante t_1, o novo endereço é válido, isto é, cada entrada de endereço está em um nível lógico válido. Neste momento, os circuitos internos da ROM começam a decodificar as novas entradas de endereço para selecionar o registrador que vai mandar seus dados para os buffers de saída. No instante t_2, a entrada $\overline{CS}$ é ativada para habilitar os buffers de saída. Finalmente, em t_3, as saídas mudam do estado de alta impedância para valores de dados válidos que representam os dados armazenados no endereço especificado.

O intervalo de tempo compreendido entre t_1, quando o novo endereço se torna válido, e t_3, quando os dados na saída se tornam válidos, é chamado de tempo de acesso t_{ACC}. ROMs bipolares típicas têm um tempo de acesso na faixa de 30 a 90 ns. Os tempos de acesso de dispositivos NMOS se encontram na faixa de 35 a 500 ns. Os avanços na tecnologia CMOS trouxeram os tempos de acesso para uma faixa

entre 20 e 60 ns. Conseqüentemente, as ROMs mais novas raramente são produzidas como dispositivos NMOS ou bipolares.

Um outro parâmetro de temporização importante é o *tempo de habilitação da saída*, t_{OE}, que representa o atraso existente entre a entrada $\overline{CS}$ e a saída de dados válidos. Os valores típicos para t_{OE} são 10 a 20 ns para bipolares, 25 a 100 para NMOS e 12 a 50 ns para CMOS. Este parâmetro de temporização é importante em situações em que as entradas de endereço já estão nos seus novos valores, mas as saídas da ROM ainda não foram habilitadas. Quando $\overline{CS}$ vai para BAIXO para habilitar as saídas, o atraso será igual a t_{OE}.

11-7 TIPOS DE ROMs

Agora que temos uma compreensão geral da arquitetura interna e da operação externa de uma ROM, vamos estudar os diversos tipos de ROMs para analisar as diferenças no modo em que elas são programadas, apagadas e reprogramadas.

ROM Programada por Máscara

A ROM programada por máscara tem suas posições escritas (programadas) pelo fabricante de acordo com as especificações do cliente. Um negativo fotográfico chamado *máscara* é usado para especificar as conexões elétricas no chip. Uma máscara específica é necessária para cada conjunto de informações diferentes a ser armazenado na ROM. Uma vez que essas máscaras são muito caras, esse tipo de ROM só é economicamente viável se uma grande quantidade de uma mesma ROM é necessária. Algumas ROMs desse tipo estão disponíveis como dispositivos pré-programados com informações ou dados normalmente usados, como por exemplo códigos geradores de caracteres usados em terminais de vídeo. A principal desvantagem desse tipo de ROM é que ela não pode ser reprogramada quando uma mudança de projeto necessitar que os dados armazenados sejam modificados. A ROM teria que ser trocada por uma outra com os novos dados gravados. Vários tipos de ROMs programáveis pelo usuário foram desenvolvidos para superar essa limitação. Entretanto, ROMs programadas por máscara ainda representam a alternativa mais econômica quando uma grande quantidade de ROMs com programação idêntica é necessária.

ROMs programadas por máscara são comumente chamadas simplesmente de ROMs. Entretanto, isso pode trazer alguma confusão, uma vez que o termo ROM na verdade representa uma categoria mais abrangente de dispositivos que durante a sua operação normal podem ser apenas lidos. Usaremos o acrônimo MROM (*mask-programmed* ROM) sempre que fizermos referência a ROMs programadas por máscara.

A Fig. 11-9 mostra a estrutura de uma pequena MROM MOS. Ela consiste em 16 células de memória organizadas em quatro linhas de quatro células. Cada célula é um transistor MOSFET canal-N na configuração dreno comum (entrada na porta e saída na fonte). A linha superior das células (LINHA 0) constitui um registrador de 4 bits. Observe que alguns dos transistores nessa linha (Q_0 e Q_2) têm suas fontes conectadas na linha de saída da coluna, enquanto outros (Q_1 e Q_3) não têm. O mesmo acontece com as células de outras linhas. A presença ou ausência dessas conexões nos terminais fontes dos transistores determina se o valor armazenado na célula é igual a um ou zero, respectivamente. A conexão de cada terminal fonte é controlada durante a produção por uma máscara fotográfica baseada nos dados fornecidos pelo cliente.

Observe que as saídas de dados estão conectadas nas colunas. Em relação à saída D_3, por exemplo, qualquer transistor que possui uma conexão do terminal fonte com a coluna de saída (como Q_0, Q_4 e Q_8), pode comutar V_{dd} para a coluna, colocando-a em estado lógico ALTO. Se V_{dd} não for conectado na coluna, a saída será mantida em nível lógico BAIXO por um resistor de pull-down. Em qualquer instante, apenas um transistor da coluna deve ser colocado em estado de condução pelo decodificador da linha.

O decodificador de 2 para 4 é usado para decodificar as entradas de endereço, A_1A_0, e selecionar de que linha (registrador) os dados devem ser lidos. As saídas ativas em ALTO do decodificador fornecem os sinais de habilitação, que, por sua vez, estão conectados nos terminais de entrada (portas) de várias células. Se a entrada de habilitação do decodificador $\overline{EN}$ é mantida em ALTO, todas as saídas estarão no estado inativo BAIXO, e os transistores não estarão conduzindo porque não haverá tensão na porta. Para esta situação, as saídas de dados estarão em nível BAIXO.

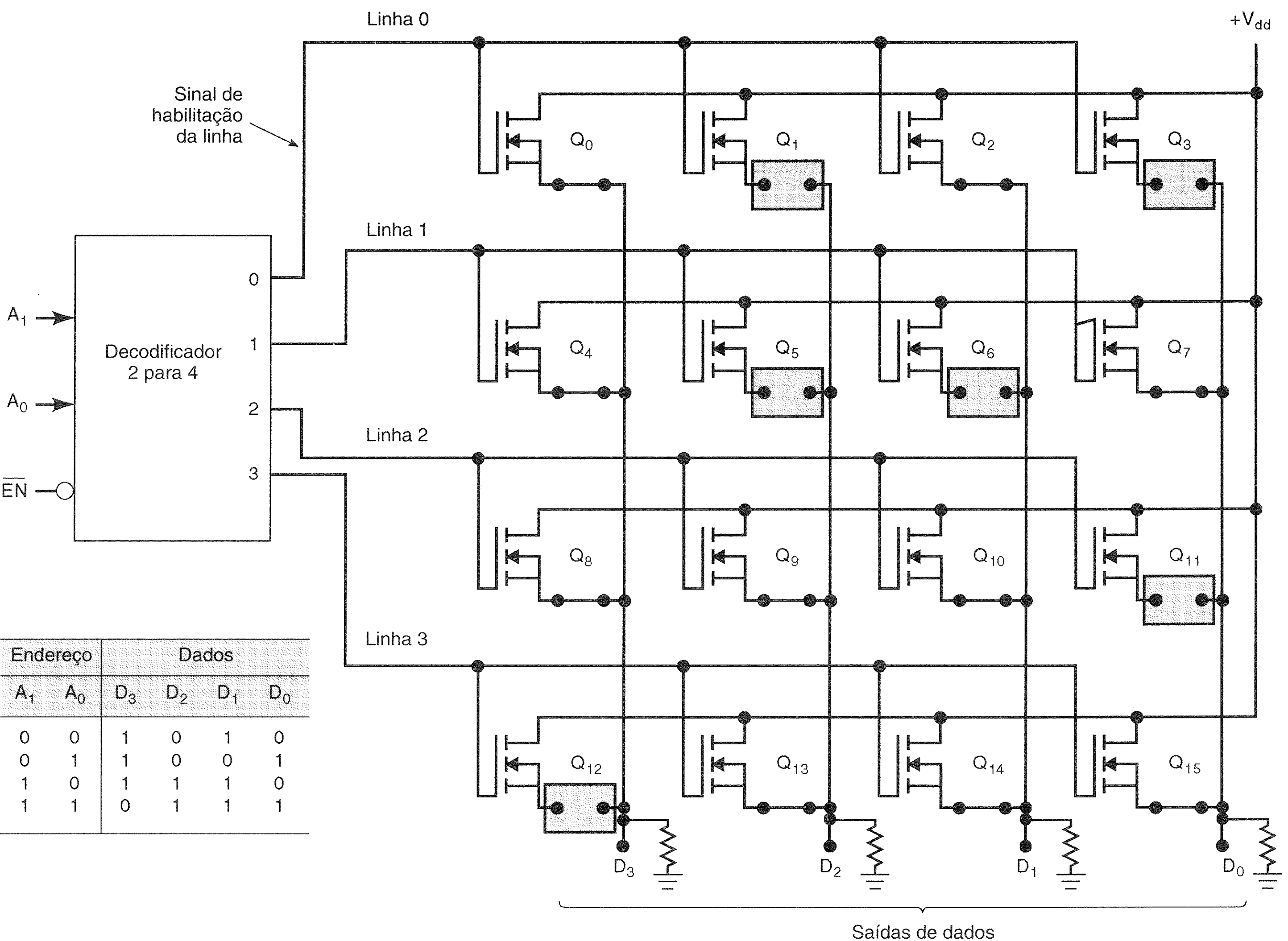
Quando $\overline{EN}$ está em BAIXO, os níveis presentes nas entradas de endereço determinam que linha (registrador) estará habilitada, de modo que seus dados possam ser lidos nas saídas. Por exemplo para ler a LINHA 0, as entradas A_1A_0 devem estar em 00. Isto coloca um nível ALTO em LINHA 0. Todas as outras linhas estarão em 0 V. Este nível ALTO em LINHA 0 faz com que os transistores Q_0, Q_1, Q_2 e Q_3 conduzam. Com todos os transistores da linha conduzindo, V_{dd} aparecerá no terminal fonte de cada transistor. As saídas D_3 e D_1 irão para ALTO, uma vez que Q_0 e Q_2 estão conectados nas suas respectivas colunas. As saídas D_2 e D_0 permanecerão em BAIXO porque não existe uma conexão entre os terminais fontes dos transistores Q_1 e Q_3 e suas respectivas colunas. De maneira semelhante, outros endereços farão com que os dados do registrador correspondente apareçam nas saídas. A tabela na Fig. 11-9 mostra os dados correspondentes para cada endereço. Você deve verificar como isso está relacionado com as conexões do terminal fonte de várias células.

EXEMPLO 11-8

As MROMs podem ser usadas para armazenar tabelas de funções matemáticas. Mostre como a MROM na Fig. 11-9 pode ser usada para armazenar a função $y = x^2 + 3$, onde as entradas de endereço fornecem o valor para x e o dado de saída fornece o valor para y.

Solução

O primeiro passo é construir uma tabela relacionando a saída desejada para cada uma das entradas. O número binário, x,



Endereço		Dados			
A_1	A_0	D_3	D_2	D_1	D_0
0	0	1	0	1	0
0	1	1	0	0	1
1	0	1	1	1	0
1	1	0	1	1	1

Fig. 11-9 A estrutura de uma MROM MOS mostra que um MOSFET é usado para cada célula de memória. Uma conexão aberta do terminal fonte armazena "0"; uma conexão fechada armazena "1".

TABELA 11-1

x		$y = x^2 + 3$			
A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	1	1
0	1	0	1	0	0
1	0	0	1	1	1
1	1	1	1	0	0

é representado pelo endereço A_1A_0. O número binário de saída é o valor desejado para y. Por exemplo, quando $x = A_1A_0 = 10_2 = 2_{10}$, a saída deve ser igual a $2^2 + 3 = 7_{10} = 0111_2$. A Tabela 11-1 mostra todos os valores de saída para cada um dos valores de entrada possíveis. Esta tabela é fornecida ao fabricante da MROM para desenvolvimento de uma máscara que indicará as conexões apropriadas entre as células de memória durante o processo de fabricação. Por exemplo, a primeira linha da tabela indica que as conexões dos terminais fontes de Q_0 e Q_1 não devem ser feitas, enquanto as conexões para Q_2 e Q_3 devem.

MROMs bipolares têm uma estrutura similar à Fig. 11-9, exceto que as células são transistores bipolares, em vez de MOSFETs. A TMS47256 é uma versão NMOS que tem uma capacidade de 32K × 8. Seu símbolo pode ser visto na Fig. 11-10. Note que ela possui saídas tristate para permitir uma interface simples com o barramento de dados de um computador. Além das catorze entradas de endereço, ela possui duas entradas de habilitação $\overline{E}$ e $\overline{S}$. Ambas devem estar em BAIXO para habilitar as saídas da MROM. A entrada $\overline{E}$ também é responsável por realizar o **power-down**. Quando $\overline{E}$ está em nível ALTO, os circuitos internos do chip são colocados em standby, consumindo apenas um quarto da corrente que consumiriam em operação normal. A TMS47256 tem um tempo de acesso de 200 ns e um consumo de potência em standby de 82,5 mW. A versão CMOS, a TMS47C256, tem um tempo de acesso de 100 ns e consumo em standby de apenas 2,8 mW.

ROMs Programáveis (PROMs — *Programmable* ROMs)

Uma ROM programável por máscara é muito cara e só seria usada em aplicações que necessitassem de um grande volume, pois o alto custo poderia ser diluído pelas várias unidades. Para aplicações de volume mais baixo, os fabricantes desenvolveram PROMs com **conexões a fusível** que são programadas pelo usuário, isto é, não são programadas durante o processo de fabricação, mas são programadas pelo usuário, de acordo com suas necessidades. Entretanto, uma vez programada, uma PROM é como uma MROM, e não pode ser apagada e reprogramada. Logo, se um programa em uma PROM tem erros ou deve ser alterado, esta PROM deve ser "jogada fora". Por essa razão, esses dispositivos são também chamados de ROMs programáveis apenas uma vez (OTP — *one time programmable*).

A estrutura de uma PROM com conexão a fusível é bastante similar à estrutura da MROM devido ao fato de que algumas conexões podem ser deixadas intactas ou abertas para programar a célula com 1 ou 0, respectivamente. Na MROM da Fig. 11-9, essas conexões eram entre o terminal fonte e a coluna de saída. Na PROM, cada uma dessas conexões é feita através de um fusível que vem intacto do fabricante (ver a Fig. 11-11).

O usuário pode seletivamente *queimar* qualquer um desses fusíveis para produzir um determinado conjunto de dados armazenados na memória. Tipicamente, um valor pode ser programado ou "queimado" em um determinado endereço do seguinte modo: fornece-se o endereço nas entradas de endereço; coloca-se o valor que deve ser programado nos terminais de dados; e então aplica-se V_{pp}, um pulso de alta tensão (10-30 V), em um pino especial de programação do CI. A Fig. 11-11 mostra a operação de programação. Todos os transistores na linha selecionada (linha 0) são colocados em estado de condução, e V_{pp} está aplicado nos seus drenos. Nas colunas (linhas de dados) onde foi colocado um nível lógico 0 (por exemplo, Q_1), uma corrente alta passará pelo fusível, fazendo com que ele se abra e o nível 0 fique permanentemente armazenado. As colunas que possuem nível lógico 1 (por exemplo, Q_0) têm V_{pp} em uma extremidade do fusível e V_{dd} na outra. Isto faz com que a corrente que passa pelo fusível seja bem menor, deixando o fusível intacto. Uma vez que todas as células tenham sido programadas, como explicado anteriormente, os dados estão armazenados de modo permanente na PROM e podem ser lidos repetidamente acessando o endereço apropriado. O valor dos dados armazenados não será alterado se a alimentação do chip for retirada, porque nada fará com que um fusível aberto seja reconstituído.

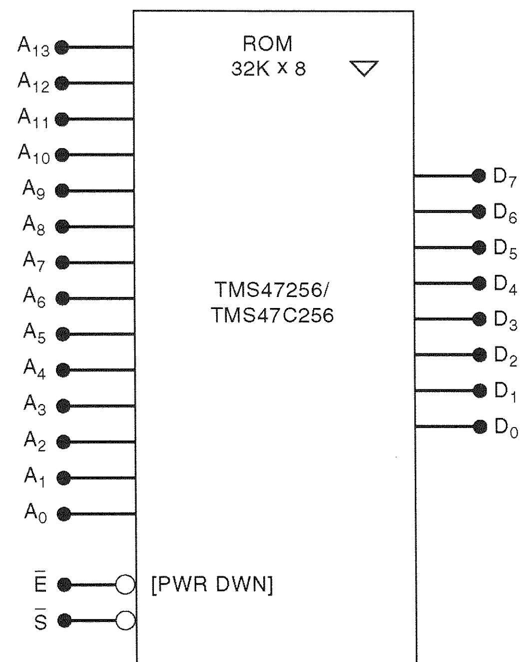


Fig. 11-10 Símbolo lógico para a MROM TMS47256 fabricada com tecnologia NMOS/CMOS.

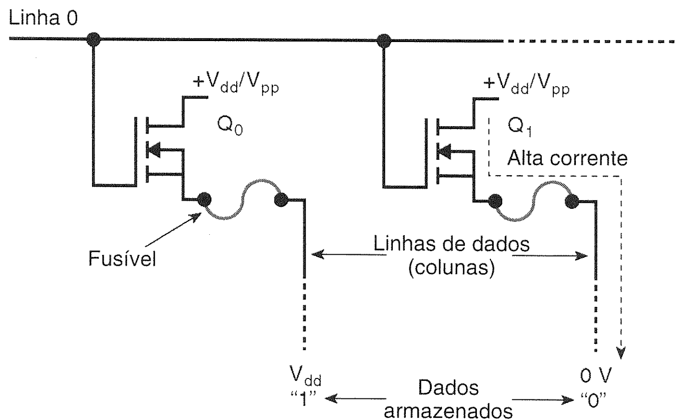


Fig. 11-11 As PROMs utilizam fusíveis que podem ser abertos seletivamente pelo usuário para programar o nível lógico 0 na célula.

O processo de programação de uma PROM e a verificação de que os dados armazenados estão corretos raramente são feitos de modo manual. Em vez disso, esse processo é feito de modo automático por um equipamento chamado de **programador** de PROM. Tipicamente, a PROM é colocada em um soquete do programador. Os circuitos do programador selecionam cada endereço da PROM, programam ("queimam") os dados corretos para este endereço, verificam se os dados gravados estão corretos e passam para o próximo endereço para repetir o processo. Os dados que serão programados na PROM são fornecidos ao programador através do teclado, ou de uma unidade de disco, ou ainda transferidos de um computador. Esta última operação é chamada de *downloading* e permite que o usuário desenvolva e teste o programa, ou os dados a serem armazenados, e depois quando essa etapa estiver terminada, o usuário possa transferir esses dados da memória do computador para o programador de PROM, que por sua vez irá gravá-los na PROM.

Muito poucas PROMs bipolares ainda estão disponíveis atualmente. Hoje em dia, quase todas usam tecnologia MOS, e a CMOS está se tornando a mais popular. A TMS27PC256 é uma PROM CMOS bastante popular, com capacidade de 32K × 8 e consumo de potência em standby de apenas 1,4 mW. Está disponível com tempos de acesso máximos variando de 100 a 250 ns.

ROM Programável e Apagável (*Erasable Programmable* ROM — EPROM)

Uma EPROM pode ser programada pelo usuário e também pode ser *apagada* e reprogramada tão freqüentemente quanto desejado. Uma vez programada, a EPROM é uma memória *não-volátil*, que irá manter indefinidamente os dados armazenados. O processo de programação de uma EPROM envolve a aplicação de níveis de tensão especiais (geralmente na faixa de 10 a 25 V) nas entradas do chip apropriadas, por um determinado intervalo de tempo (geralmente 50 ms para cada posição). O processo de programação geralmente é realizado por um circuito de programação especial que está separado do circuito que será utilizado na EPROM durante sua operação normal. O processo de programação completo pode durar até vários minutos para um chip EPROM.

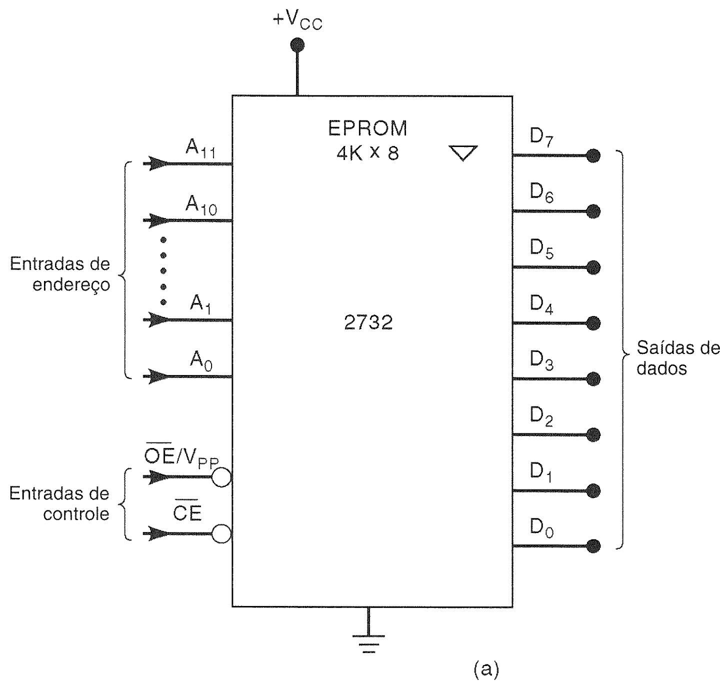
As células de armazenamento em uma EPROM são transistores MOS cuja porta não possui conexão elétrica (isto é, a porta está em *flutuação*). Em seu estado normal, cada transistor não está conduzindo e cada célula está armazenando um nível lógico 1. Um transistor pode ser colocado em um estado de condução aplicando-se um pulso de programação de alta tensão que injeta elétrons com alta energia na região da porta em flutuação. Os elétrons permanecem presos nesta região tão logo o pulso é retirado, porque não existe um caminho de descarga. Isto mantém o transistor em estado de *permanente* condução mesmo quando a alimentação é removida do dispositivo, e a célula agora está armazenando um nível lógico 0. Durante o processo de programação, os pinos de dados e endereços da EPROM são usados para selecionar quais as células de memória serão programadas com 0s e quais serão mantidas com 1s.

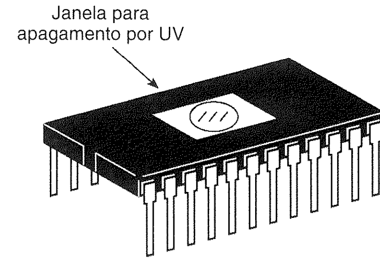
Uma vez programada a célula de uma EPROM, ela pode ser apagada sendo exposta à luz ultravioleta (UV) aplicada pela janela existente no encapsulamento do chip. A luz ultravioleta produz uma corrente da porta em flutuação para o substrato de silício, que remove as cargas armazenadas e faz com que o transistor passe para um estado de não-condução, restabelecendo o nível lógico 1. Este processo de apagamento geralmente requer de 15 a 20 minutos de exposição à luz ultravioleta. Infelizmente, não existe um modo de apagar apenas algumas células selecionadas; *a luz ultravioleta apaga todas as células ao mesmo tempo*, de modo que uma EPROM apagada armazena apenas 1s. Uma vez apagada, ela pode ser reprogramada.

As EPROMs estão disponíveis em um amplo espectro de capacidades e tempos de acesso. Dispositivos com capacidade de 512K × 8 e tempos de acesso de 120 ns são comuns. A 2732 é um exemplo de uma pequena EPROM. A 2732 é uma EPROM NMOS de 4K × 8 que opera com uma fonte única de +5 V durante operação normal. A Fig. 11-12(a) mostra o símbolo lógico para a 2732. Observe que existem 12 entradas de endereço, uma vez que $2^{12} = 4.096$ e 8 saídas de dados. Ela possui duas entradas de controle. $\overline{CE}$ é a entrada de habilitação do chip e é utilizada para colocar o dispositivo em modo standby, onde o consumo de potência é reduzido. $\overline{OE}/V_{PP}$ é uma entrada cuja função irá depender do modo de operação do dispositivo. $\overline{OE}$ é o sinal de habilitação da saída e é usado para controlar os buffers de saída do dispositivo, de modo que ele possa ser conectado ao barramento de dados de um microprocessador sem que haja contenção de barramento. V_{PP} é a tensão de programação especial necessária durante o processo de programação.

O encapsulamento da 2732, que aparece na Fig. 11-12(b), mostra a "janela" que permite que os circuitos internos sejam expostos à luz ultravioleta quando for necessário apagar todo o conteúdo da memória. Uma fita adesiva opaca é colocada sobre a janela após a reprogramação para evitar que ela seja acidentalmente apagada pela luz ambiente.

A 2732 possui diversos modos de operação controlados pelas tensões aplicadas nos pinos $\overline{CE}$ e $\overline{OE}/V_{PP}$, conforme mostra a Fig. 11-12(c). O modo de programação é usado para escrever novos dados nas células da EPROM. Isto é feito





(b)

	Entradas		
Modo	$\overline{CE}$	$\overline{OE}/V_{PP}$	Dados
LEITURA/VERIFICAÇÃO	V_{IL}	V_{IL}	Saída de dados
INIBIÇÃO da Saída	V_{IL}	V_{IH}	Alta impedância
STANDBY	V_{IH}	X	Alta impedância
PROGRAMAÇÃO	V_{IL}	V_{PP}	Entrada de dados

(c)

Nota: V_{IL} = nível BAIXO TTL
V_{IH} = nível ALTO TTL
X = don't care
V_{PP} = 21 V nominal

Fig. 11-12 (a) Símbolo lógico para a EPROM 2732; (b) encapsulamento típico de uma EPROM mostrando a janela utilizada para apagamento com luz ultravioleta; (c) modos de operação da 2732.

geralmente em uma EPROM "limpa", isto é, uma que foi previamente apagada com luz ultravioleta e que está com todas as suas células armazenando nível lógico 1. O processo de programação escreve uma palavra de oito bits em um endereço de cada vez, conforme se segue: (1) o endereço é fornecido aos pinos de endereço; (2) os dados que devem ser programados são colocados nos pinos de dados, que funcionam como entradas durante o processo de programação; (3) uma tensão alta, nominalmente de 21 V, é aplicada a V_{PP}; e (4) $\overline{CE}$ é pulsado em nível BAIXO, geralmente por 50 ms. Este processo é repetido para todas as posições de memória. Se ele fosse feito manualmente, poderia levar várias horas. Geralmente, entretanto, ele é feito automaticamente com um programador de EPROM disponível no mercado, bastante semelhante aos programadores de PROM descritos anteriormente. Uma EPROM apagada pode ser programada em alguns minutos, tão logo os dados a serem programados tenham sido transferidos para o programador de EPROM.

A 27C512 da Intel é uma EPROM de 64K × 8 que pode ser programada muito mais rapidamente que a 2732. A 27C512 necessita que o pulso em $\overline{CE}$ seja de apenas 100 μs para escrever um byte, em vez de um pulso de 50 ms necessário para a 2732. Isto significa que o tempo total de programação fica em um intervalo entre 8 e 10 segundos. A 27C512, como todas as EPROMs, é apagada por exposição à luz ultravioleta durante 15 a 20 minutos.

As EPROMs foram projetadas originalmente para pesquisa e desenvolvimento de aplicações, em que a necessidade de alterar o programa armazenado diversas vezes é bastante comum. À medida que se tornaram mais confiáveis e baratas, elas se transformaram em uma opção atraente para serem utilizadas em aplicações de baixo e médio volumes. Hoje em dia, milhões de EPROMs ainda estão em uso. Entretanto, elas possuem diversas desvantagens que foram superadas pelas EEPROMs e dispositivos de memória FLASH, por isso não estão sendo muito utilizadas em novos projetos. Suas principais desvantagens são: (1) devem ser removidas do circuito para serem apagadas e reprogramadas; (2) a operação de apagamento apaga o chip inteiro, não há meio de selecionar apenas determinados endereços que devem ser apagados; (3) apagar e reprogramar um chip demora cerca de 20 minutos ou mais.

PROM Apagável Eletricamente (*Electrically Erasable* PROM — EEPROM*)

As desvantagens da EPROM foram superadas pelo desenvolvimento da **PROM apagável eletricamente (EEPROM)** como um aperfeiçoamento da EPROM. A EEPROM mantém a mesma estrutura de porta em flutuação da EPROM, mas com a adição de uma fina camada de óxido acima do dreno do MOSFET da célula de memória. Essa modificação produz a principal característica da EEPROM, isto é, poder ser apagada eletricamente. Aplicando-se uma tensão alta (21 V) entre a porta e o dreno do MOSFET, uma carga é induzida na porta em flutuação, onde permanecerá mesmo que a alimentação seja retirada; a aplicação reversa da mesma tensão faz com que as cargas que estavam na porta em

* Também denominadas E²PROM. (*N.T.*)

flutuação sejam removidas e que a célula seja apagada. Uma vez que esse mecanismo de transporte de cargas necessita de correntes muito baixas, o apagamento e a programação de uma EEPROM podem ser feitos *no próprio circuito*, sem necessitar de uma fonte de luz ultravioleta e de um programador especial.

Uma outra vantagem da EEPROM em relação à EPROM é a capacidade de apagar e reescrever bytes *individuais* na matriz de memória. Durante a operação de escrita, os circuitos internos apagam as células, que estão localizadas no endereço fornecido, antes de escreverem os novos dados. Essa capacidade de apagar os bytes individualmente torna bem mais simples a tarefa de modificar os dados armazenados na EEPROM. Além disso, uma EEPROM pode ser programada mais rapidamente do que muitas EPROMs. Tipicamente, uma operação de escrita em um determinado endereço dura cerca de 5 ms, o que é bem menos do que os 50 ms necessários para uma EPROM. Entretanto, EPROMs mais novas são bem mais rápidas (100 μs).

As primeiras EEPROMs, como a 2816 da Intel, necessitavam de circuitos de suporte externo aos chips de memória. Estes incluíam um conversor DC-DC que gerava a tensão de programação, V_{PP}, de 21 V a partir de uma fonte de +5 V e um circuito para controlar a temporização e a seqüência das operações de apagamento e programação. Os dispositivos mais novos, como a 2864 da Intel, têm esses circuitos de suporte integrados no mesmo chip que a matriz de memória, de modo que necessitam apenas de uma fonte única de +5 V. Isto torna as EEPROMs tão fáceis de utilizar quanto as memórias de leitura/escrita que iremos estudar em breve.

A capacidade de apagar os bytes individuais em uma EEPROM e o seu alto nível de integração são conseguidos à custa da densidade e do custo. A complexidade da célula de memória e a colocação dos circuitos de suporte no chip colocam as EEPROMs bem atrás das EPROMs em termos de capacidade de armazenamento por milímetro quadrado de silício. Uma EEPROM de 1 megabit necessita de aproximadamente duas vezes mais silício do que uma EPROM de 1 megabit. Portanto, apesar da sua superioridade funcional, suas desvantagens em termos de densidade e custo têm evitado que elas substituam as EPROMs em aplicações onde a densidade e o custo são fatores importantes.

O símbolo lógico para a 2864 da Intel pode ser visto na Fig. 11-13(a). Ela está organizada em um arranjo de 8K × 8 com 13 entradas de endereço ($2^{13} = 8.192$) e oito pinos de entrada e saída de dados. Três entradas de controle determinam o modo de operação de acordo com a tabela dada na Fig. 11-13(b). Quando $\overline{CE}$ = ALTO, o chip está em modo standby (baixo consumo), nenhuma operação é realizada em qualquer posição de memória e os pinos de dados estão em alta impedância.

Para que uma posição de memória seja lida, o endereço da posição desejada deve ser aplicado aos pinos de endereço. $\overline{CE}$ é colocado em BAIXO e o pino de habilitação da saída, $\overline{OE}$, é colocado em nível BAIXO, habilitando os buffers da saída de dados. O pino que habilita a escrita, $\overline{WE}$, é mantido em ALTO durante a operação de leitura.

Para que uma posição de memória seja escrita (programada), os buffers da saída devem ser desabilitados para que o dado que será escrito possa ser aplicado como entrada nos pinos de entrada/saída. A temporização de uma operação de escrita pode ser vista na Fig. 11-13(c). Até o instante t_1, o dispositivo está no modo standby. Um novo endereço é fornecido neste instante. Em t_2, as entradas $\overline{CE}$ e $\overline{WE}$ são colocadas em nível BAIXO para iniciar uma operação de escrita. A entrada $\overline{OE}$ está em ALTO, e, portanto, os pinos de dados permanecerão em alta impedância. O dado é aplicado aos pinos de entrada e saída em t_3 e é escrito na posição selecionada pelo endereço na subida de $\overline{WE}$ em t_4. O dado é retirado dos pinos de entrada em t_5. Na verdade, o dado é armazenado em um registrador que faz parte do circuito interno da 2864. O dado é mantido neste registrador até que outros circuitos realizem a operação de apagamento da posição de memória selecionada, após o que o byte de dados é transferido do registrador e armazenado na posição de memória. O processo de apagar e armazenar dura cerca de 5 ms. Quando $\overline{CE}$ retorna ao nível ALTO em t_4, o chip retorna ao modo standby, enquanto as operações de apagamento e armazenamento são completadas.

A 2864 tem um modo de escrita aperfeiçoado que permite que o usuário escreva até 16 bytes em um armazenamento temporário até que os circuitos da EEPROM apaguem as posições de endereço selecionadas. Esses 16 bytes são transferidos então para a matriz da EEPROM para serem armazenados nessas posições. Esse processo também dura cerca de 5 ms.

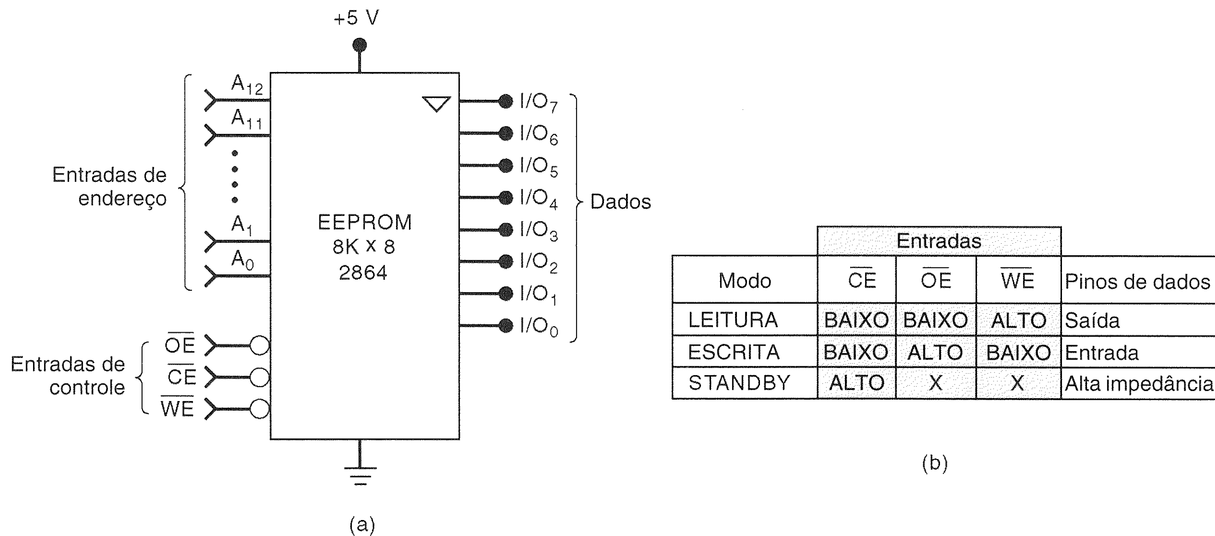
Como o processo interno de armazenamento de dados na EEPROM é bastante lento, a velocidade de transferência desses dados também pode ser bastante baixa. Conseqüentemente, muitos fabricantes oferecem dispositivos EEPROM em encapsulamento de oito pinos que são interfaceados a um barramento *serial* de dois ou três fios. Isso poupa espaço físico em uma placa de circuito impresso, ao contrário do que acontece quando se usa uma 2864 em um encapsulamento DIP de 28 pinos. Isso também simplifica a interface entre a CPU e a EEPROM.

CD ROM

Um tipo de armazenamento somente para leitura bastante utilizado em computadores hoje são os CDs (*compact disks*). A tecnologia de fabricação e os circuitos necessários para recuperar a informação são os mesmos daqueles utilizados em sistemas de áudio. Apenas o formato dos dados é diferente. Os CDs são fabricados com uma superfície altamente reflexiva. Para armazenar dados nesses discos, um feixe laser bastante intenso é focalizado sobre um ponto *muito* pequeno no disco. Esse feixe faz uma depressão que é capaz de provocar a difração da luz nesse ponto da superfície. Os dados digitais (1s e 0s) são armazenados no disco, um bit por vez, fazendo ou não essa depressão no material reflexivo. A informação digital no disco está organizada em uma espiral contínua de pontos de dados. A precisão do feixe laser permite que grandes quantidades de dados (acima de 550 Mbytes) sejam armazenados em um pequeno disco de 120 mm.

Para que os dados possam ser lidos, um feixe laser bem menos potente é focalizado na superfície do disco. Em qualquer ponto, a luz refletida é sentida como um 1 ou um 0. Esse arranjo ótico é montado em um mecanismo de transporte que se move para a frente e para trás ao longo do

raio do disco, seguindo a espiral de dados à medida que o disco roda. Os dados recebidos do sistema ótico vêm em uma linha serial de dados, um bit por vez. A rotação do disco é controlada para manter uma taxa constante de bits. Se esse disco está sendo usado para reprodução de áudio, essa linha de dados é convertida em formato analógico. Se esse disco está sendo usado como uma memória somente de leitura, esses dados são codificados em bytes paralelos que o computador pode usar. A tecnologia do CD, embora seja bastante sofisticada, é relativamente barata e está se tornando um meio padronizado para carregar grandes quantidades de dados em um computador. Os principais avanços que estão sendo feitos nessa tecnologia se referem a melhorias do tempo de acesso na recuperação dos dados.



	Entradas			
Modo	$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	Pinos de dados
LEITURA	BAIXO	BAIXO	ALTO	Saída
ESCRITA	BAIXO	ALTO	BAIXO	Entrada
STANDBY	ALTO	X	X	Alta impedância

(b)

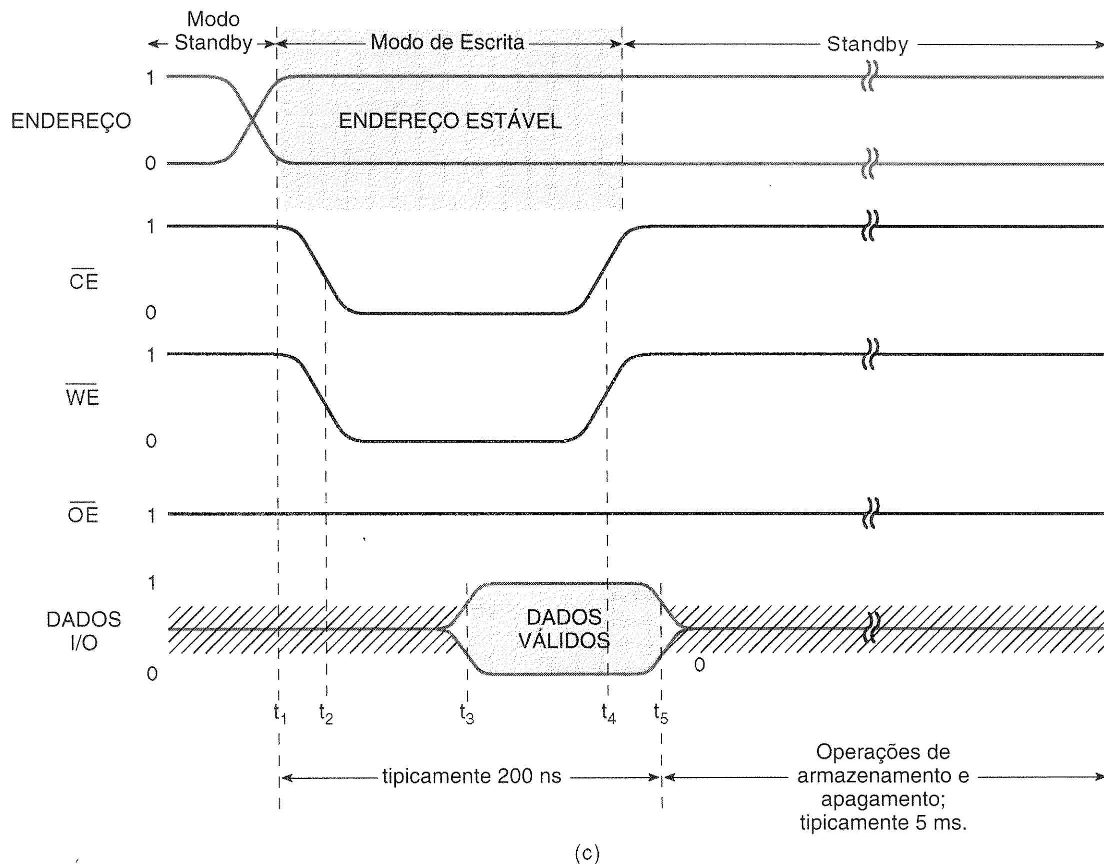


Fig. 11-13 (a) Símbolo para a EEPROM 2864; (b) modos de operação; (c) temporização para a operação de escrita.

Questões de Revisão

1. *Verdadeiro ou falso*: Uma MROM pode ser programada pelo usuário.
2. Em que uma PROM difere de uma MROM? Ela pode ser apagada e reprogramada?
3. *Verdadeiro ou falso*: Uma PROM armazena nível lógico "1" quando seu fusível está intacto.
4. Como uma EPROM é apagada?
5. *Verdadeiro ou falso*: Não existe modo de apagar apenas uma parte de uma memória EPROM.
6. Que função é realizada pelos programadores de PROM e de EPROM?
7. Quais são as desvantagens das EPROMs que foram superadas pelas EEPROMs?
8. Quais as maiores desvantagens das EEPROMs?
9. Em que tipo de ROM se pode apagar um byte de cada vez?
10. Quantos bits são lidos de um CD ROM em qualquer instante de tempo?

11-8 MEMÓRIA FLASH

As EPROMs são não-voláteis, oferecem tempos de acesso reduzidos (tipicamente 120 ns) e têm alta densidade e baixo custo por bit. Entretanto, elas precisam ser retiradas do circuito/sistema para serem apagadas e reprogramadas. As EEPROMs são não-voláteis, oferecem tempos de acesso reduzidos e permitem que o apagamento e a reprogramação de bytes individuais sejam feitos no circuito/sistema. Entretanto, apresentam como desvantagens a baixa densidade e o alto custo quando comparadas às EPROMs.

O desafio para os engenheiros de semicondutores era fabricar uma memória não-volátil, apagável eletricamente como a EEPROM, mas com densidades e custos semelhantes aos da EPROM, ainda mantendo a alta velocidade do acesso de ambas. A resposta a esse desafio foi a **memória flash**.

Estruturalmente, uma célula de memória flash é semelhante à célula com um único transistor da EPROM (não é semelhante à célula mais complexa da EEPROM com dois transistores), sendo apenas ligeiramente maior. Ela possui uma camada de óxido mais fina na porta que permite que a célula possa ser apagada eletricamente, permitindo, ainda, que se obtenham densidades bem maiores do que as observadas nas EEPROMs. O custo de uma memória flash é consideravelmente menor do que uma EEPROM, embora ainda não esteja tão próximo do custo da EPROM. Essa diferença de custo deverá diminuir à medida que a tecnologia flash seja aperfeiçoada. A Fig. 11-14 mostra as vantagens e desvantagens dos vários tipos de memórias semicondutoras não-voláteis. À medida que aumenta a flexibilidade para apagar e programar, também aumentam a complexidade e o custo do dispositivo. A MROM e a PROM são os dispositivos mais simples e baratos, mas não podem ser apagados e reprogramados. A EEPROM é o dispositivo mais complexo e caro porque pode ser apagado e programado "byte a byte".

As memórias flash são assim chamadas porque possuem tempos curtos de apagamento e escrita. A maioria dos chips faz *apagamento em bloco*, nos quais todas as células no chip são apagadas simultaneamente. Esse processo de apagamento em bloco dura, tipicamente, centenas de milissegundos, muito menos tempo do que os 20 minutos necessários para as EPROMs. Algumas memórias flash mais novas oferecem um *modo de apagamento por setor*, em que setores específicos da matriz de memória (por exemplo, grupos de 512 bytes) podem ser apagados de uma só vez. Isso evita que seja necessário apagar e reprogramar todas as células quando apenas uma parte da memória tiver que ser atualizada. Uma memória flash típica possui um tempo de escrita de 10 μs por byte. Compare esse tempo com os 100 μs das EPROMs mais avançadas e com os 5 ms da EEPROM (que inclui o tempo necessário para apagar um byte).

Os fabricantes de memória estão trabalhando no desenvolvimento e aperfeiçoamento de dispositivos de memória flash, e cada um tem os seus próprios critérios para determinar quais as capacidades e quais as características de

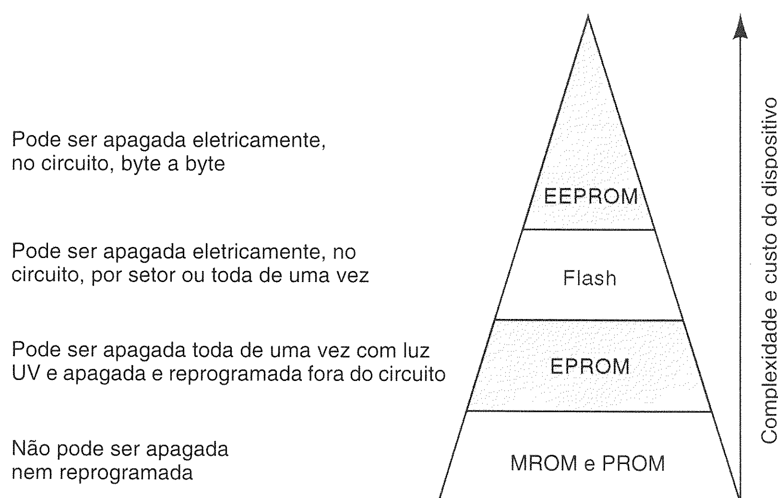
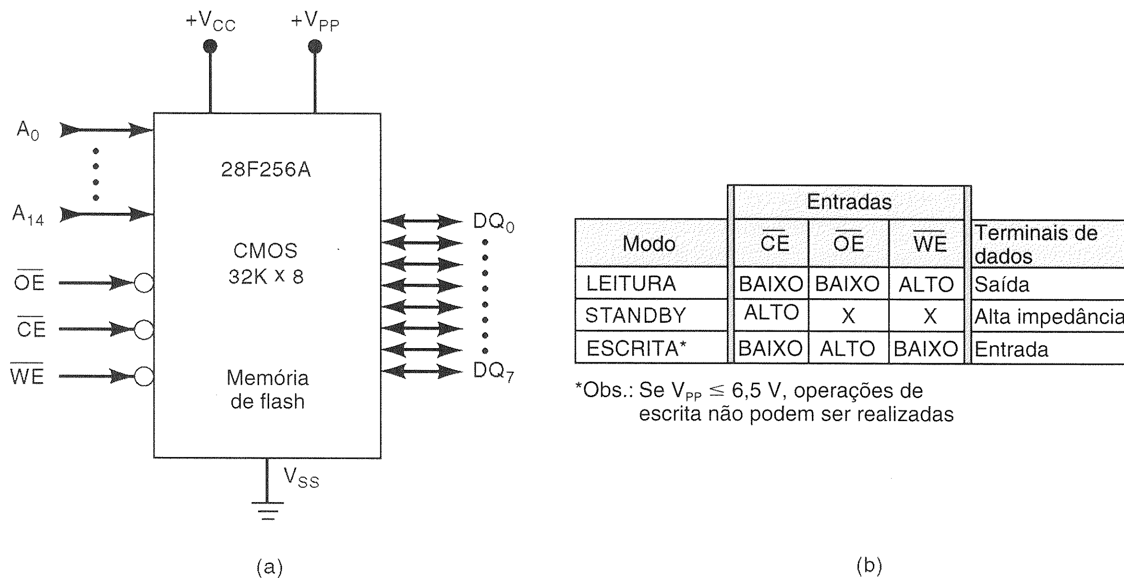


Fig. 11-14 As relações de compromisso entre as memórias semicondutoras não-voláteis mostram que a complexidade e o custo aumentam à medida que a flexibilidade no apagamento e na programação aumenta.



Modo	Entradas $\overline{CE}$	$\overline{OE}$	$\overline{WE}$	Terminais de dados
LEITURA	BAIXO	BAIXO	ALTO	Saída
STANDBY	ALTO	X	X	Alta impedância
ESCRITA*	BAIXO	ALTO	BAIXO	Entrada

*Obs.: Se $V_{PP} \leq 6{,}5$ V, operações de escrita não podem ser realizadas

Fig. 11-15 (a) Símbolo lógico para o chip de memória flash 28F256A; (b) entradas de controle $\overline{CE}$, $\overline{WE}$ e $\overline{OE}$.

desempenho mais importantes. Não tentaremos pesquisar aqui um grande número de fabricantes e dispositivos. Em vez disso, apresentaremos um CI de memória flash representativo de um dos fabricantes líderes na fabricação de memórias e vamos usá-lo como um veículo para aprender os aspectos principais da operação da memória flash. Analisando esse componente representativo, aprenderemos a usar *códigos de comando* como uma maneira de controlar operações internas de chips lógicos mais complexos. Isso servirá como uma boa apresentação a vários tipos de CIs que são controlados por comandos e que fazem parte da maioria dos sistemas baseados em microprocessadores.

O CI de Memória Flash CMOS 28F256A

A Fig. 11-15(a) mostra o símbolo lógico do chip de memória flash CMOS 28F256A da Intel, que possui 32K × 8. O diagrama mostra 15 entradas de endereço ($A_0 - A_{14}$) necessárias para selecionar as diferentes posições de memória, isto é, $2^{15} = 32\text{K} = 32.768$. Os oito pinos de entrada e saída de dados ($DQ_0 - DQ_7$) são usados como entradas durante as operações de escrita na memória e como saída durante as operações de leitura. Esses pinos de dados ficam em alta impedância quando o chip não está selecionado ($\overline{CE}$ = ALTO) ou quando as saídas estão desabilitadas ($\overline{OE}$ = ALTO). A entrada de habilitação de escrita ($\overline{WE}$) é usada para controlar as operações de escrita na memória. Note que esse chip necessita de duas tensões de alimentação: V_{CC}, que é a fonte padrão de +5 V usada pelos circuitos lógicos; V_{PP}, que é a tensão de alimentação para apagar/programar, nominalmente +12 V, que é necessária para as operações de apagamento e programação (escrita).

As entradas de controle ($\overline{CE}$, $\overline{OE}$ e $\overline{WE}$) controlam o que acontece com os pinos de dados de modo bem semelhante àquele descrito para a EEPROM 2864, como mostra a tabela na Fig. 11-15(b). Esses pinos de dados normalmente estão conectados ao barramento de dados. Durante uma operação de escrita, os dados são transferidos pelo barramento, geralmente do microprocessador, para o chip. Durante uma operação de leitura, o dado que está dentro do chip é transferido para o barramento, geralmente para o microprocessador. É importante observar que, se o pino de V_{PP} não for mantido em uma tensão alta (acima de 6,5 V), as operações de escrita não podem ser realizadas, e o chip só pode ser lido; portanto, ela funciona como uma ROM cujo conteúdo não pode ser alterado.

A operação de um chip de memória flash pode ser mais bem compreendida estudando-se a sua estrutura interna. A Fig. 11-16 é um diagrama da 28F256A mostrando seus principais blocos funcionais. Você deve voltar a esse diagrama sempre que necessário durante o estudo que faremos a seguir. Uma característica singular dessa estrutura é a existência de um *registrador de comandos*, que é usado para gerenciar todas as funções do chip. Os códigos de comando (ou simplesmente comandos) são escritos nesse registrador para controlar as funções que acontecem no chip (por exemplo, apagar, apagar-verificar, programar, programar-verificar). Esses comandos geralmente vêm pelo barramento de dados, oriundos do microprocessador. A lógica de controle de estados examina o conteúdo do registrador de comandos e gera os sinais lógicos e de controle para outros circuitos do chip para que estes executem as etapas necessárias de uma operação. Vamos estudar como isso funciona para alguns comandos.

Comando de Leitura

Para preparar o chip para operações de leitura, é necessário *escrever* todos os 0s ($00000000_2 = 00_{16}$) no registrador de comandos. Isto é feito colocando-se 00_{16} nos pinos de dados, pulsando $\overline{WE}$ em BAIXO, enquanto $\overline{CE}$ = BAIXO e $\overline{OE}$ = ALTO [conforme mostra a tabela na Fig. 11-15(b)]. As entradas de endereço podem estar com qualquer nível porque os dados de entrada são dirigidos automaticamente

para o registrador de comandos. Uma vez feito isso, o dispositivo está pronto para *ler* os 262.144 bits (32.768 bytes) da matriz de memória. Os dados da memória são lidos de modo usual: (1) é fornecido o endereço da posição a ser lida; (2) $\overline{WE}$ é colocado em ALTO, $\overline{CE}$ em BAIXO, e $\overline{OE}$ é pulsado em BAIXO para habilitar os buffers de saída a passar o conteúdo da posição acessada para os pinos de saída de dados. O tempo de acesso para leitura em uma 28F256A é de no máximo 120 ns. O dispositivo permanece habilitado para leitura enquanto o conteúdo do registrador for igual a 00_{16}.

Comandos Apagar e Preparar para Apagar

Para apagar todo o conteúdo da memória, são necessários dois passos: (1) escrever 20_{16} no registrador de comandos para preparar o chip para a operação de apagar e (2) escrever 20_{16} no registrador de comandos novamente para iniciar a operação de apagar. Essa seqüência de dois passos dificulta que o conteúdo da memória seja apagado acidentalmente. Após o segundo comando ter sido escrito, a tensão 12 V de V_{PP} é usada para apagar todas as células. Esse processo dura cerca de 10 ms.

Comando Verificar o Apagamento

Após a operação de apagar, é necessário verificar se todas as células de memória foram apagadas, isto é, todos os bytes = 11111111_2 = FF_{16}. A verificação do apagamento deve ser feita para cada posição de memória. Ela é iniciada escrevendo-se $A0_{16}$ no registrador de comandos. Logo a seguir, é realizada uma operação de leitura no endereço a ser verificado, e o valor lido é comparado com FF_{16}. Este dois passos são feitos para cada endereço, e o tempo total de verificação é da ordem de 1 s. Se o conteúdo da posição de memória não for igual a FF, o chip deve ser apagado novamente usando o comando de apagar descrito anteriormente.

Comandos Preparação para Programar e Programação

Um dispositivo pode ser preparado para a programação de um byte escrevendo-se o código 40_{16} no registrador de comandos. Logo a seguir, o dado a ser programado é escrito na posição desejada.

Comando Verificação de Programação

Após um byte ter sido programado em um endereço, o conteúdo deste endereço deve ser verificado para assegurar que essa posição foi programada corretamente. Isso é feito, primeiramente, escrevendo-se o código $C0_{16}$ no registrador de comandos para preparar a operação de verificação. A isso se segue uma operação de leitura, fazendo com que o chip forneça como saída o conteúdo do endereço que acabou de ser programado para que, então, este possa ser comparado com o valor que se desejava programar.

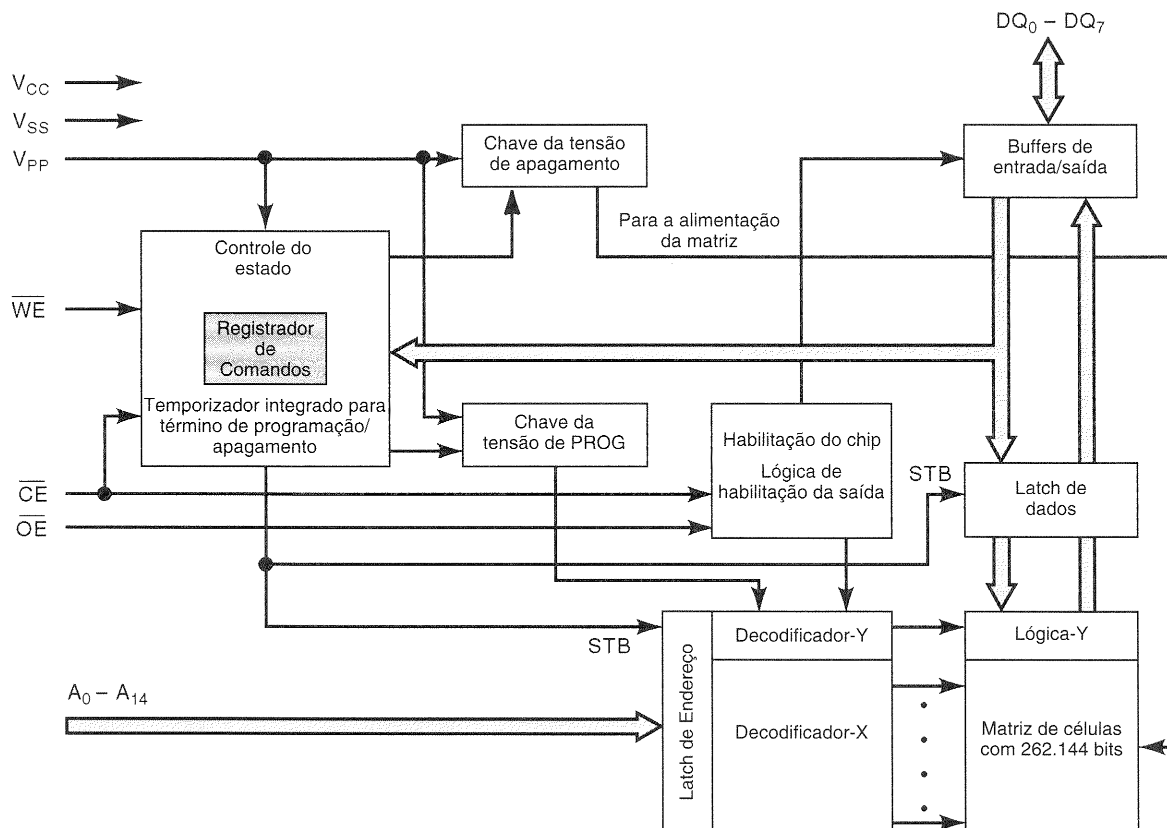


Fig. 11-16 Diagrama funcional do chip de memória flash 28F256A. (Cortesia: Intel Corporation.)

O processo completo de programação e verificação de cada endereço leva cerca de 500 ms para a 28F256A.

Questões de Revisão

1. Qual a principal vantagem da memória flash sobre as EPROMs?
2. Qual a principal vantagem da memória flash sobre as EEPROMs?
3. Qual o significado da palavra *flash*?
4. Para que serve V_{PP}?
5. Qual a função do registrador de comandos da 28F256A?
6. Qual o objetivo do comando de verificação do apagamento?
7. Qual o objetivo do comando de verificação da programação?

11-9 APLICAÇÕES DAS ROMS

Com exceção das MROMs e PROMs, a maioria dos dispositivos ROM pode ser reprogramada, então tecnicamente eles não são memórias "somente de leitura". Entretanto, o termo *ROM* ainda pode ser usado para EPROMs, EEPROMs e memórias flash, porque durante a operação normal o conteúdo armazenado não é alterado tão freqüentemente quanto é lido. Portanto, o termo ROM pode ser aplicado a todos os dispositivos de memória semicondutora não-volátil, e são usados em aplicações em que o armazenamento não-volátil de informações, dados ou programas é necessário e os dados armazenados raramente ou nunca mudam. A seguir falaremos sobre as áreas de aplicação mais comuns.

Firmware

A mais difundida aplicação das ROMs é o armazenamento de dados e programas que devem estar disponíveis no processo de inicialização de sistemas microprocessados. Esses dados e programas são chamados de **firmware** porque estão "firmemente armazenados no hardware" (isto é, em chips ROM) e não estão sujeitos a alterações durante a operação normal do sistema. Alguns computadores pessoais (PCs), computadores comerciais e laptops armazenam seus sistemas operacionais e interpretadores de linguagens (por exemplo, BASIC, Pascal) no firmware, ficando portanto armazenados em ROM, de modo que o computador possa usá-los imediatamente após ter sido ligado.

Além dos microcomputadores, existem muitos outros sistemas baseados em microprocessadores que utilizam firmware. Produtos de consumo como automóveis, videocassetes, CD players e fornos de microondas, bem como diversos tipos de máquinas industriais, possuem microcontroladores dedicados, que por sua vez são compostos de um microprocessador que controla e monitora a operação de acordo com os programas e dados que estão armazenados na ROM (firmware).

Em todos esses sistemas, a operação é bastante dependente do que está armazenado na ROM. Alterações na operação do sistema são geralmente feitas alterando-se o conteúdo da ROM. No caso das EPROMs, é necessário que ou se removam esses componentes do sistema para que sejam apagados e reprogramados ou que esses componentes sejam trocados por outros. Isso faz com que haja uma interrupção na operação do sistema que pode variar de alguns minutos até uma hora. O uso de EEPROMs, ou, melhor ainda, de memórias flash, diminui consideravelmente essa interrupção, pois elas podem ser rapidamente apagadas e reprogramadas no próprio sistema.

Memória de Bootstrap

Muitos microcomputadores e a maioria dos computadores maiores não têm seu sistema operacional armazenado em ROM. Em vez disso, esses programas estão armazenados em um dispositivo externo de memória de massa, geralmente um disco magnético. Então como esses computadores sabem o que fazer quando são ligados? Um programa relativamente pequeno chamado **programa de bootstrap*** está armazenado na ROM. Quando o computador é ligado, ele executa as instruções que estão nesse programa de bootstrap. Essas instruções geralmente inicializam o hardware do sistema. O programa de bootstrap então carrega o sistema operacional para a memória interna principal. Nesse ponto, o computador começa a executar o sistema operacional e está pronto a responder aos comandos do usuário. Este processo de inicialização é freqüentemente chamado de "boot" do sistema.

Tabelas de Dados

As ROMs são bastante usadas para armazenar tabelas de dados que não variam. Alguns exemplos são as tabelas trigonométricas (isto é seno, cosseno etc.) e tabelas de conversão de código. O sistema digital pode usar essas tabelas para "procurar" o valor correto. Por exemplo, uma ROM poderia ser usada para armazenar a função seno para ângulos entre 0° e 90°. Ela poderia ser organizada em uma matriz 128 × 8 com sete entradas de endereços e oito bits de dados. As entradas de endereço representam o ângulo em incrementos de 0,7°. Por exemplo, o endereço 0000000 representa 0°, o endereço 0000001 representa 0,7°, o endereço 0000010 representa 1,41°, e assim por diante, até o endereço 11111111, que é 89,3°. Quando o endereço é fornecido à ROM, as saídas de dados representarão o seno aproximado do ângulo. Por exemplo, para o endereço 1000000 (que representa aproximadamente 45°), as saídas serão 10110101. Uma vez que o seno de um ângulo é menor ou igual a 1, este valor deve ser interpretado como uma fração, isto é, 0,10110101, que, quando convertido para decimal, é igual a 0,707 (o seno de 45°). É fundamental que o usuário dessa ROM compreenda o formato nos quais os dados estão armazenados.

Antigamente, ROMs com tabelas para funções como aquela vista anteriormente eram disponíveis como chips TTL. Hoje, apenas algumas ainda são produzidas. A maioria dos sistemas que precisa procurar valores equivalentes possui um microprocessador, e a tabela onde esses valores são ar-

* Bootstrap significa tira ou correia de bota; o nome é usado por causa da expressão *to raise oneself by one's own bootstraps*, que significa fazer-se por si próprio, subir por seu próprio esforço. (*N.T.*)

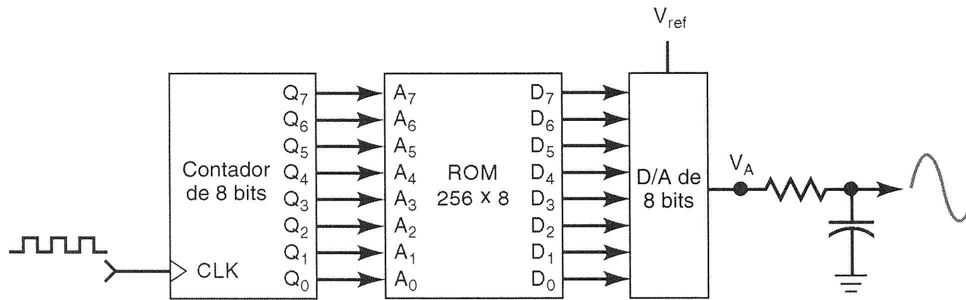


Fig. 11-17 Gerador de funções usando uma ROM e um D/A.

mazenados está na mesma ROM que contém as instruções do programa.

Conversor de Códigos

Um circuito conversor de códigos toma valores expressos em um determinado tipo de código e produz uma saída expressa em outro código. A conversão de códigos é necessária, por exemplo, quando um computador está fornecendo como saída dados em formato binário simples e queremos convertê-los para BCD para que estes possam ser mostrados em um display de 7 segmentos.

Um dos métodos mais simples de conversão de código usa uma ROM programada de modo que a aplicação de um endereço particular (o código antigo) produz uma saída que o representa em um novo código. A 74185 é uma ROM TTL que armazena a tabela de conversão de código binário-BCD para um número de entrada de seis bits. Por exemplo, o endereço de entrada binário 100110 (decimal 38) vai produzir como saída 00111000, que é o código BCD para o decimal 38. O Problema 11-20 vai utilizar esta ROM.

Gerador de Funções

Um circuito gerador de funções é um circuito que produz formas de onda como senóides, ondas triangulares e quadradas. A Fig. 11-17 mostra como uma tabela armazenada em ROM e um conversor D/A são usados para gerar uma senóide.

A ROM armazena 256 diferentes valores de oito bits, cada um correspondendo a um valor diferente da forma de onda, isto é, um ponto de tensão diferente na senóide. O contador de oito bits é continuamente incrementado pelo sinal de clock para fornecer endereços à ROM de modo seqüencial. À medida que o contador cicla pelos 256 endereços diferentes, a ROM fornece como saída os 256 pontos para o D/A. A saída do D/A será uma forma de onda que terá 256 níveis de tensão diferentes correspondentes aos dados armazenados na ROM. Um filtro passa-baixa suaviza os degraus na saída do D/A.

Circuitos como esses são usados em alguns geradores de função comerciais. A mesma idéia é aplicada a sintetizadores de voz, em que valores digitalizados da forma de onda da voz são armazenados em ROM. O ML2035, mostrado na Fig. 11-18, é um gerador programável de ondas senoidais que incorpora essa estratégia de gerar uma senóide de amplitude fixa e uma freqüência que pode ser selecionada entre DC e 50 kHz. O número que é colocado no registrador de deslocamento de 16 bits é usado para determinar a freqüência do clock para o contador que aciona as entradas de endereço da ROM. O ML2035 foi projetado para aplicações em telecomunicações que necessitam que diversas freqüências sejam geradas com precisão.

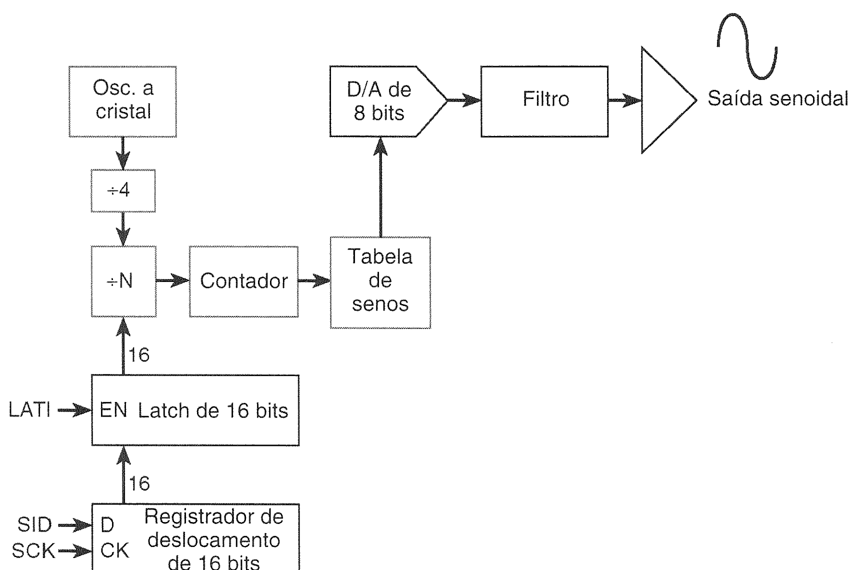


Fig. 11-18 Gerador de senóides programável. (Cortesia: Microlinear.)

Armazenamento Auxiliar

Por serem não-voláteis, de alta velocidade, com baixo consumo e não possuírem partes móveis, módulos de memória flash estão se tornando alternativas aos discos magnéticos de armazenamento. Isto é verdade, em particular, para baixas capacidades (5 Mbytes ou menos). Nessa faixa, a memória flash consegue competir com os discos magnéticos. O baixo consumo de potência a torna especialmente atrativa para computadores portáteis do tipo laptop e notebook que utilizam alimentação a bateria. Podemos esperar ver mais e mais desses "discos semicondutores", isto é, placas com vários CIs de memória flash e lógica de controle, usados no lugar de discos mais lentos, volumosos, menos confiáveis e de maior consumo de potência.

Questões de Revisão

1. O que é firmware?
2. Descreva como um computador usa um programa de bootstrap.
3. O que é um conversor de código?
4. Quais são os principais elementos de um gerador de funções?
5. Por que módulos de memória flash são uma alternativa viável para discos de armazenamento auxiliar?

11-10 DISPOSITIVOS DE LÓGICA PROGRAMÁVEL (PLDs)

Os projetistas de circuitos lógicos têm à sua disposição uma ampla faixa de CIs padronizados, com diversas funções e circuitos lógicos em chips. Além disso, esses CIs são fabricados por várias empresas a um custo relativamente baixo. Por essas razões, os projetistas interconectam CIs padronizados para formar uma variedade quase interminável de diferentes circuitos e sistemas, e continuarão a fazer isso indefinidamente no futuro.

Entretanto, existem alguns problemas com circuitos e sistemas que utilizam apenas os CIs padronizados. Alguns projetos de sistemas podem necessitar de centenas ou milhares de CIs. Esse grande número de CIs necessita de um espaço considerável em uma placa de circuito impresso e uma grande quantidade de tempo colocando, soldando e testando esses CIs. O fabricante do equipamento deve manter um estoque considerável de todos os CIs utilizados no projeto. Reduzindo o número de CIs usados em um projeto podemos conseguir diversas vantagens: um menor espaço em placa, e portanto um menor número de placas de circuito impresso e gabinetes menores; um menor consumo de potência (menores fontes de alimentação); processos de fabricação mais rápidos e baratos; maior confiabilidade, uma vez que existe um menor número de CIs e de conexões que podem falhar; e uma manutenção mais fácil.

Para reduzir o número de CIs a serem usados no projeto, é necessário colocar mais e mais funções em um chip. É claro que isso tem sido feito com as tecnologias LSI e VLSI para funções padronizadas como memórias, microprocessadores, sintetizadores de voz, chips de calculadoras e assim por diante. Esses dispositivos contêm centenas e milhares de portas lógicas conectadas para operar de um modo predeterminado. Existem muitas situações, entretanto, para as quais não existem soluções LSI e VLSI. Estas situações têm obrigado os projetistas a interconectar diversos chips MSI e SSI para obter as funções necessárias. O desenvolvimento recente dos **dispositivos lógicos programáveis**, ou **de lógica programável**, (**PLDs**), ofereceu aos projetistas uma alternativa para substituir um grande número de CIs padronizados por um único CI.

Um PLD é um CI que contém um grande número de portas, FFs, e registradores que estão interconectados no chip. Muitas dessas conexões são fusíveis que podem ser queimados. Dizemos que um CI é *programável* porque a função específica do CI para uma dada aplicação pode ser determinada selecionando-se que conexões devem ser abertas e quais devem ficar intactas. O processo de "queima de fusíveis" pode ser feito pelo fabricante de acordo com as instruções do cliente, ou pode ser feito pelo próprio cliente. Esse processo é chamado de **programação** porque produz o padrão desejado de interconexão de portas, FFs, registradores e assim por diante.

Idéia Básica

A Fig. 11-19 demonstra a idéia básica utilizada pelos CIs de lógica programável. Ela mostra uma matriz de portas AND e uma matriz de portas OR, sendo que esta última pode ser usada para gerar quatro saídas, cada uma das quais pode ser qualquer função lógica das duas variáveis de entrada A e B.

Cada uma das entradas é aplicada em um buffer inversor e em outro não-inversor, para produzir a forma normal e a invertida de cada variável de entrada. Estas são as *linhas de entrada* para a matriz de portas AND. Cada porta AND é conectada a duas linhas de entrada diferentes para gerar um produto único das variáveis de entrada. As saídas das portas AND são chamadas de *linhas de produto*.

Cada uma das linhas de produto está conectada a uma das quatro entradas de cada uma das portas OR através de um fusível. Com todos os fusíveis intactos, as saídas das portas OR estarão sempre em 1. Aqui está a prova:

$$\begin{aligned} O_1 &= \overline{A}\,\overline{B} + \overline{A}B + A\overline{B} + AB \\ &= \overline{A}(\overline{B} + B) + A(\overline{B} + B) \\ &= \overline{A} + A = 1 \end{aligned}$$

Cada uma das saídas O_1, O_2, O_3 e O_4 pode ser *programada* para implementar qualquer função de A e B, queimando-se seletivamente os fusíveis apropriados. Os PLDs são projetados de modo que uma entrada OR com um fusível aberto aja como um nível lógico 0. Por exemplo, se queimarmos os fusíveis 1 e 4 na porta OR 1, a saída O_1 torna-se

$$O_1 = 0 + \overline{A}B + A\overline{B} + 0 = \overline{A}B + A\overline{B}$$

Podemos programar cada uma das saídas das portas OR para qualquer função desejada de maneira semelhante. Uma vez que todas as saídas tenham sido programadas, o dispositivo irá gerar permanentemente as funções selecionadas.

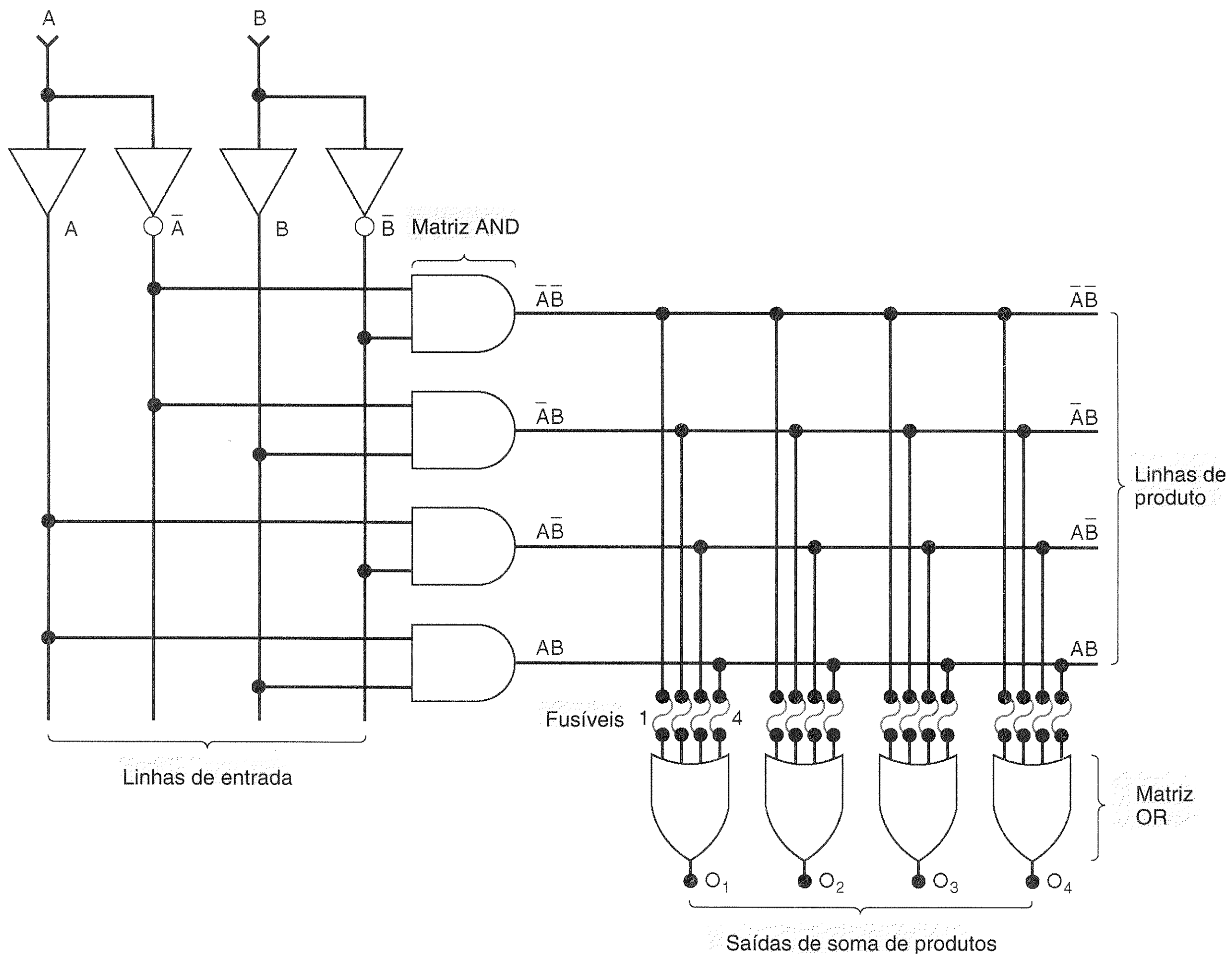


Fig. 11-19 Exemplo de um dispositivo lógico programável.

Simbologia Utilizada em PLDs

O exemplo na Fig. 11-19 tem apenas duas variáveis, e o diagrama do circuito já fica bastante difícil de entender. Você pode imaginar como seria confuso um diagrama para um PLD com um número muito maior de entradas. Por essa razão, os fabricantes de PLDs adotaram uma simbologia simplificada para representar os circuitos internos desses dispositivos.

A Fig. 11-20 mostra um exemplo dessa simbologia para uma porta AND de quatro entradas. Primeiro, observe que os buffers de entrada são representados como um único buffer com duas saídas. Além disso, observe que uma linha única é mostrada indo para a porta AND para representar as quatro entradas. As conexões das linhas de entrada das variáveis para a porta AND são indicadas por um *X* ou por um ponto. Um *X* representa que o fusível está intacto. Um ponto representa uma conexão permanente. A ausência de um desses símbolos indica que não há conexão. Nesse exemplo, as entradas *A* e $\bar{B}$ estão conectadas na porta AND para gerar o produto $A\bar{B}$. As entradas $\bar{A}$ e *B* não estão conectadas na porta AND (não há *X* ou ponto), portanto não têm efeito sobre a saída da porta AND. O que é importante para ser entendido sobre essa simbologia é que a porta AND possui *quatro* entradas diferentes mesmo que apenas uma única linha seja mostrada. As saídas que de fato estão conectadas na porta AND estão simbolizadas por *X* e/ou por ponto.

Arquiteturas de PLD — a PROM

Diversas arquiteturas comuns são usadas para PLDs. A primeira que iremos examinar é a PROM. A Fig. 11-21(a) mos-

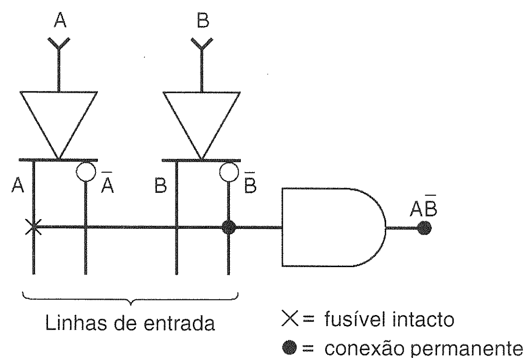


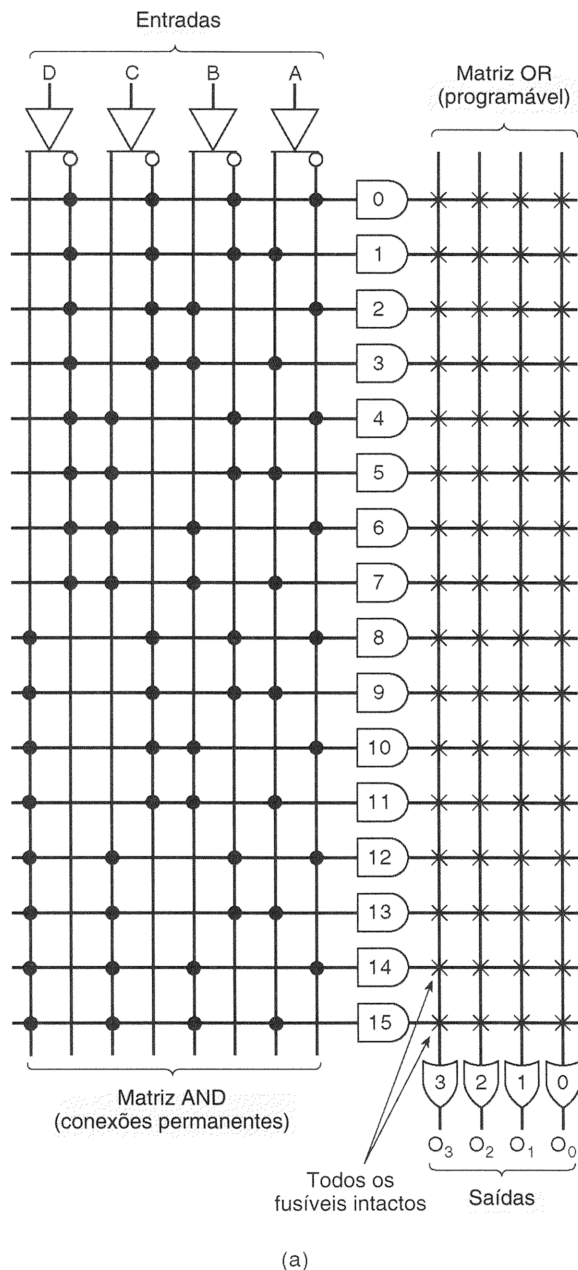
Fig. 11-20 Simbologia simplificada para PLDs.

tra uma PROM de 16 × 4 que também pode funcionar como um PLD. Ela possui quatro entradas que são completamente decodificadas por uma matriz AND, isto é, cada porta AND gera um dos 16 produtos possíveis. As conexões das linhas de entrada para a matriz AND são permanentes, enquanto as conexões das linhas de produto para as entradas das portas OR são todas programáveis.

A Fig. 11-21(b) mostra como uma PROM seria programada para gerar quatro funções lógicas especificadas. Vamos seguir o procedimento para a saída $O_3 = AB + \bar{C}\ \bar{D}$. O primeiro passo é desenhar a tabela-verdade mostrando todos os níveis da saída O_3 para todas as possíveis combinações de entrada (Tabela 11-2).

A seguir, escreva todos os produtos AND para os casos em que a saída deve ser igual a 1. A saída O_3 deve ser um OR de todos esses produtos. Assim, somente os fusíveis que conectam esses produtos nas entradas da porta OR 3 devem ficar intactos. Todos os outros devem ser queimados, como indicado na Fig. 11-21(b). Esse mesmo procedimento deve ser executado para determinar a condição dos fusíveis nas entradas das outras portas OR.

A PROM pode gerar qualquer função lógica possível das variáveis de entrada porque ela gera todos os produtos AND possíveis. De um modo geral, qualquer aplicação que necessita que toda combinação de entrada esteja disponível é uma boa candidata a usar uma ROM. Entretanto, PROMs se



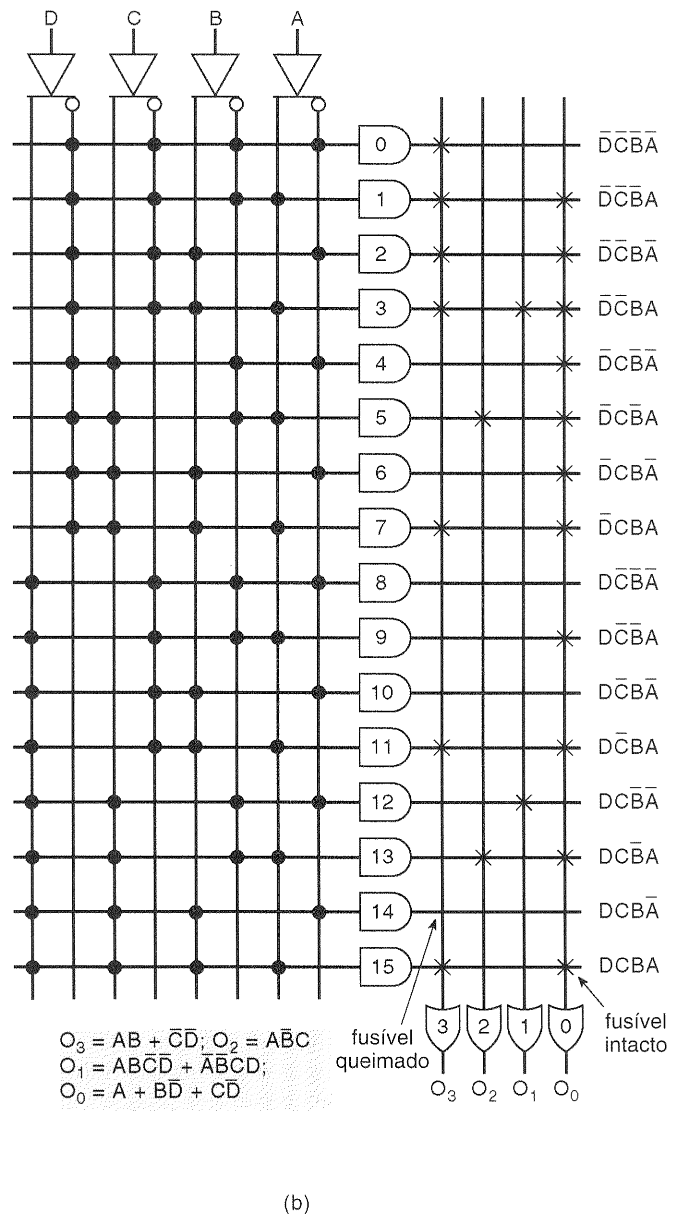


Fig. 11-21 (a) A arquitetura de uma PROM torna-a indicada para ser utilizada como um PLD; (b) fusíveis são abertos para programar as saídas para as funções desejadas.

TABELA 11-2

D	C	B	A	O_3		
0	0	0	0	1	→	$\bar{D}\,\bar{C}\,\bar{B}\,\bar{A}$
0	0	0	1	1	→	$\bar{D}\,\bar{C}\,\bar{B}A$
0	0	1	0	1	→	$\bar{D}\,\bar{C}\,B\,\bar{A}$
0	0	1	1	1	→	$\bar{D}\,\bar{C}BA$
0	1	0	0	0		
0	1	0	1	0		
0	1	1	0	0		
0	1	1	1	1	→	$\bar{D}CBA$
1	0	0	0	0		
1	0	0	1	0		
1	0	1	0	0		
1	0	1	1	1	→	$D\bar{C}BA$
1	1	0	0	0		
1	1	0	1	0		
1	1	1	0	0		
1	1	1	1	1	→	$DCBA$

tornam impraticáveis quando um grande número de variáveis de entrada deve ser utilizado, porque o número de fusíveis dobra cada vez que uma variável de entrada é adicionada.

Um exemplo de uma PROM que é na verdade usada como um PLD é a AM27S13, que é uma PROM de 512 × 4 fabricada usando tecnologia Schottky de alta velocidade. Uma vez que $512 = 2^9$, essa PROM tem nove entradas de endereço e quatro saídas de dados. Assim, a AM27S13 pode ser programada para gerar quatro saídas, em que cada uma delas pode ser qualquer função lógica das nove entradas.

Arranjo de Lógica Programável (*Programmable Array Logic* — PAL)

A arquitetura PROM é indicada para aplicações em que cada combinação de entrada possível é necessária para gerar as funções de saída. Exemplos dessas aplicações são conversores de código e tabelas. Muitas aplicações, entretanto, não necessitam que todas as combinações de entrada sejam programáveis. Por exemplo, nem todas as funções lógicas mostradas na Fig. 11-21(b) usam todas as linhas de produto. Isso levou ao desenvolvimento de uma classe de PLDs chamada **arranjo de lógica programável (PAL)**, ou **arranjo lógico programável**. A arquitetura de uma PAL difere ligeiramente de uma PROM, como pode ser visto na Fig. 11-22(a).

A PAL tem as mesmas matrizes AND e OR que as PROMs; entretanto, na PAL as conexões para as entradas das portas AND são programáveis, enquanto as conexões para as entradas das portas OR são permanentes. Isso significa que cada porta AND pode ser programada para gerar qualquer produto das variáveis de entrada e seus complementos. Cada porta OR está permanentemente ligada a apenas quatro saídas das portas AND. Isso limita a função a quatro termos-produto. Se uma função necessita de mais de quatro termos-produto, ela não pode ser implementada com essa PAL; uma outra que tenha portas OR com um maior número de entradas teria que ser usada. Se precisarmos de um número menor do que quatro termos-produto, aqueles que não são necessários devem ser colocados em 0.

A Fig. 11-22(b) mostra como essa PAL é programada para gerar quatro funções lógicas especificadas. Vamos mostrar esse procedimento para a saída $O_3 = AB + \bar{C}\,\bar{D}$. Primeiro, devemos expressar a saída como uma soma OR de quatro termos, pois as portas OR possuem quatro entradas. Faremos isso colocando 0s. Então, teremos

$$O_3 = AB + \bar{C}\bar{D} + 0 + 0$$

Agora, devemos determinar como programar as entradas das portas AND 1, 2, 3 e 4, para que elas possam fornecer os termos-produto corretos para a porta OR número 3. Fazemos isso considerando um termo por vez. O primeiro termo, AB, é obtido deixando intactos os fusíveis que conectam as entradas A e B na porta AND número 1 e queimando todos os outros fusíveis dessa linha. Do mesmo modo, o segundo termo, $\bar{C}\,\bar{D}$ é obtido deixando intactos os fusíveis que conectam as entradas $\bar{C}$ e $\bar{D}$ na porta AND número 2. O terceiro termo é um 0. Um 0 lógico constante é produzido na porta AND número 3 deixando todos os fusíveis intactos. Isso produz um termo $A\bar{A}B\bar{B}C\bar{C}D\bar{D}$, que, como sabemos, é igual a 0. O quarto termo também é 0, e portanto os fusíveis das entradas da porta AND número 4 devem ficar intactos.

As entradas das outras portas AND são programadas de modo semelhante para gerar as outras funções de saída. Observe que muitas portas AND têm todos os seus fusíveis de entrada intactos, uma vez que eles são necessários para gerar 0s.

Um exemplo real de um circuito integrado PAL é a PAL18L8A da Texas Instruments, Inc. Ela é fabricada usando tecnologia Schottky de baixa potência e possui dez entradas e oito saídas. Cada porta OR está permanentemente conectada a sete saídas de portas AND, e portanto ela pode gerar funções que utilizem até sete termos. Um recurso presente nessa PAL, em particular, é que seis das oito saídas são realimentadas para a matriz AND, onde elas podem ser conectadas como entradas de qualquer porta AND. Isso é bastante útil para gerar todo tipo de lógica combinacional.

Fusível de Polaridade

Muitos PLDs incluem um recurso chamado **saída de polaridade programável**, que dá ao projetista a opção de inverter qualquer saída do dispositivo. Isto é ilustrado na Fig. 11-23, onde uma porta EX-OR e um *fusível de polaridade* foram adicionados na saída da porta OR. A porta EX-OR funciona como um inversor controlado que pode inverter ou não a saída, dependendo do estado do fusível. Quando o fusível está intacto (nível lógico 0), a porta EX-OR não vai modificar o valor da saída da porta OR. Quando o fusível estiver aberto (nível 1), a porta EX-OR inverterá a saída da porta OR.

Esse recurso de programação da polaridade de saída está disponível em diversas PALs, inclusive na GAL16V8, que estudaremos em detalhe no Cap. 12.

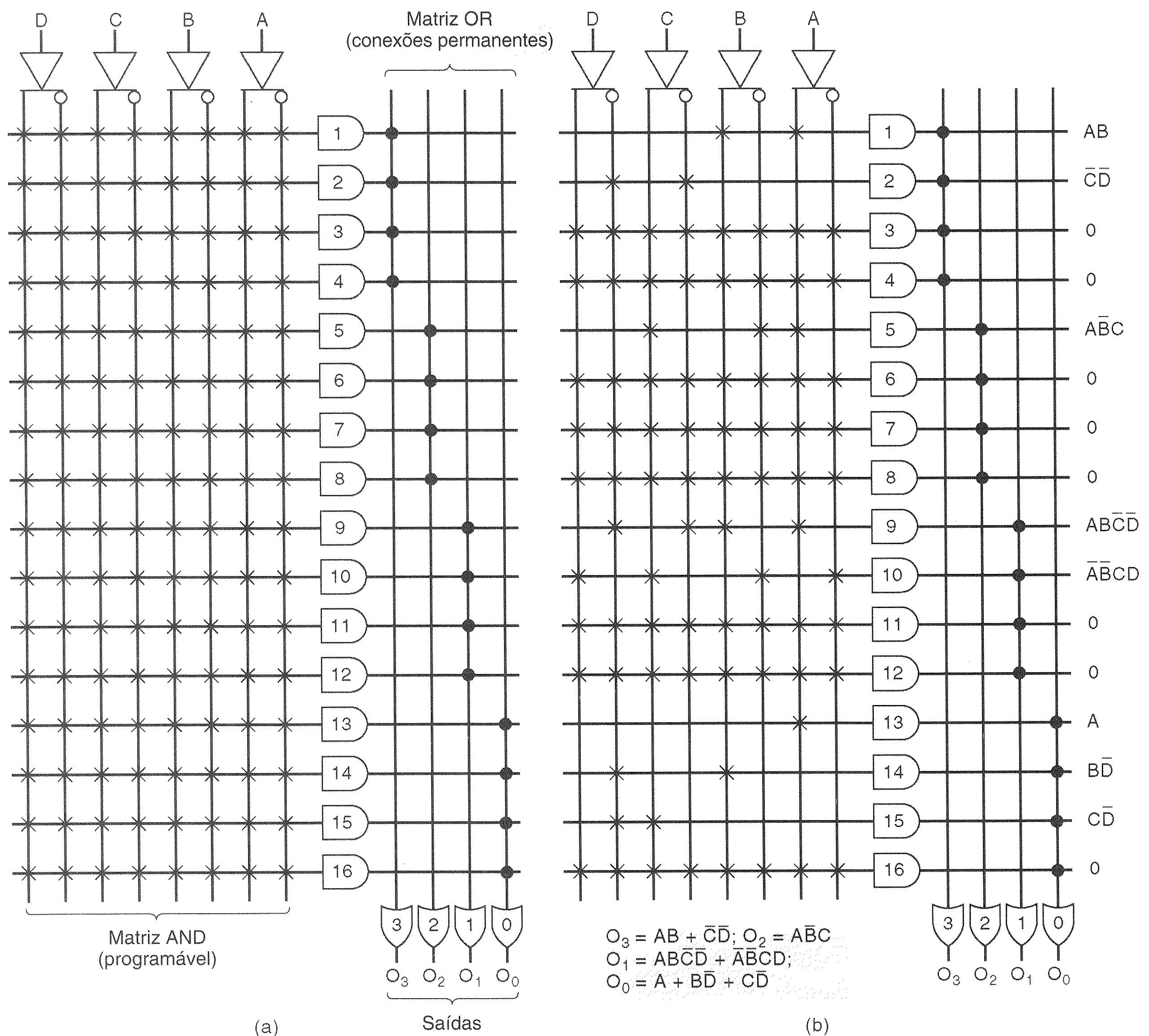


Fig. 11-22 (a) Arquitetura típica de uma PAL; (b) a mesma PAL programada para as funções desejadas.

Outros Recursos Disponíveis em PLDs

Muitos PLDs incluem um ou mais dos seguintes itens como parte de sua arquitetura: FFs, latches, registradores de entrada e registradores de saída. Freqüentemente esses recursos são programáveis, do mesmo modo que as conexões para os componentes do chip. Isso proporciona ao projetista bastante flexibilidade para o projeto de contadores e outros circuitos seqüenciais. Esse tipo de PLD é chamado de *seqüenciador lógico programável*.

Outros Tipos de PLDs

O **arranjo de lógica programável em campo (*field programmable logic array* — FPLA)** foi desenvolvido em meados da década de 1970 como o primeiro dispositivo lógico programável sem memória. Ele usava uma matriz AND programável, bem como uma matriz OR programável. Embora a FPLA seja mais flexível do que a PAL, ela não foi bem aceita pelos projetistas. As FPLAs são usadas em projetos de máquinas de estado onde um grande número de termos-produtos é necessário para cada expressão na forma de soma de produtos.

Os **PLDs complexos (CPLDs)**, também chamados *de arranjos multiníveis*, são dispositivos que combinam um número de circuitos PAL em um mesmo chip. Os próprios blocos lógicos têm conexões AND programáveis e conexões OR fixas, com um menor número de termos do que a maioria dos dispositivos PAL. Quando um maior número de produtos é necessário, um arranjo expansor NAND pode ser conectado como um termo de entrada, ou vários blocos lógicos podem ser combinados para implementar para a expressão.

Os **arranjos de portas programáveis em campo (*field programmable gate arrays* — FPGAs)** oferecem um número de blocos lógicos configuráveis que contêm lógica

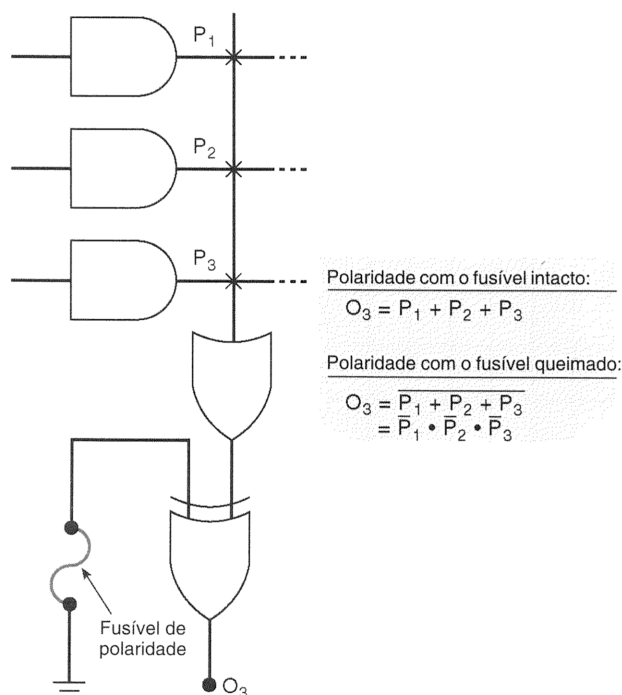


Fig. 11-23 Muitos PLDs possuem o recurso de inversão de polaridade programável que propicia a opção de inverter qualquer uma das funções de saída.

combinacional programável e registradores para circuitos seqüenciais. Além disso, existe um conjunto de blocos de entrada/saída que pode ser configurado como entrada, saída e bidirecional. As saídas são do tipo tristate, e os registradores podem ser usados para armazenar os dados de saída ou de entrada. Todos esses blocos configuráveis podem ser interconectados para implementar praticamente qualquer circuito lógico.

Obviamente, qualquer dispositivo que é programado para implementar um projeto lógico é dedicado para uma aplicação em particular. O termo **circuito integrado dedicado a uma aplicação (*application-specific integrated circuit* — ASIC)** é comumente usado para descrever um CI especificado pelo usuário que pode conter circuitos analógicos e digitais e que geralmente é programado por máscara ou fabricado sob encomenda. Muitos produtos atualmente, como por exemplo televisores, CD players e uma série de produtos na área de telecomunicações, usam ASICs para obter um alto grau de funcionalidade em encapsulamentos bem pequenos.

Programação

Quando os PLDs apareceram pela primeira vez, o projetista desenvolvia um *mapa de fusíveis* que mostrava quais fusíveis deviam ser abertos, e então enviava esses mapas para o fabricante. A partir daí, este fazia a programação do dispositivo de acordo com o mapa de fusíveis, testava e enviava os dispositivos programados de volta para o projetista. Recentemente, a disponibilidade de equipamentos de programação relativamente baratos permitiu que os próprios usuários programassem os seus PLDs. Existem no mercado programadores universais que são capazes de programar os tipos mais comuns de PROMs, PALs e FPLAs. O dispositivo a ser programado é colocado em um soquete no programador. Ele, então, programa e testa o dispositivo de acordo com os dados fornecidos pelo usuário.

A programação e as informações necessárias para teste são geralmente desenvolvidas em um software comumente disponível para rodar em PCs. Usando esse software, o usuário fornece para o computador a descrição das funções lógicas que devem ser programadas, bem como informações que indicam como o dispositivo deve ser testado. O software então gera o mapa de fusíveis e os dados de teste em um formato que pode ser enviado, por meio de um cabo, para o programador. Uma vez que o programador está com esses dados, ele pode programar e testar o dispositivo. Quando esses procedimentos estiverem terminados, o programador indicará se o dispositivo passou ou não no procedimento de teste. Se ele passar, então o dispositivo pode ser removido do soquete do programador e ser colocado no circuito protótipo para fazer outros testes.

A Fig. 11-24 ilustra um fluxograma mostrando os vários passos no processo de projeto, programação e teste de um PLD. No Cap. 12 seguiremos esse processo usando a GAL16V8 e o CUPL — um dos mais populares pacotes de software para desenvolvimento de programação de PLDs.

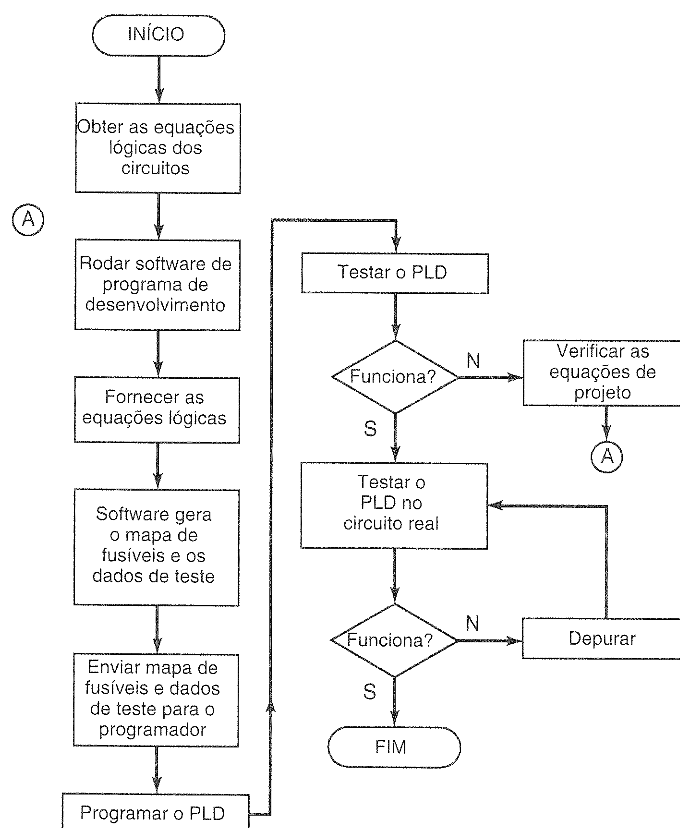


Fig. 11-24 Fluxograma do projeto, programação e teste com PLDs.

PLDs Apagáveis

Os PLDs de que estivemos falando até agora são programados queimando-se os fusíveis. Uma vez que o fusível tenha sido queimado, ele não pode ser recuperado. Então, se você cometer um erro na programação ou quiser modificar o projeto, esse dispositivo não pode ser reutilizado e portanto pode ser jogado fora. Esse problema foi estudado por diversas empresas que desenvolveram PLDs que podem ser apagados e reprogramados. Eles são chamados de *dispositivos de lógica programável apagáveis (erasable programmable logic devices— EPLDs)*, e podem ser programados e apagados do mesmo modo que as EEPROMs. A GAL16V8 é um dispositivo desse tipo. Nos PLDs apagáveis, os fusíveis são chaves eletrônicas, portanto seu estado (aberta ou fechada) pode ser eletricamente alterado quantas vezes for necessário, do mesmo modo que uma célula de memória EEPROM pode ser eletricamente apagada e reprogramada.

Questões de Revisão

1. O que é um PLD?
2. Qual seria a saída O_1 na Fig. 11-19 se os fusíveis 1 e 2 estivessem abertos?
3. O que o X representa em um diagrama de um PLD?
4. O que o ponto representa em um diagrama de um PLD?
5. Verifique se os fusíveis corretos estão abertos para as funções O_2, O_1 e O_0 na Fig. 11-21(b).
6. Como a arquitetura de uma PAL difere da de uma PROM?
7. Verifique se os fusíveis corretos estão abertos para as funções O_2, O_1 e O_0 na Fig. 11-22(b).
8. Como a arquitetura das FPLAs difere das arquiteturas de PROMs e PALs?
9. Quais são as etapas necessárias no projeto, programação e teste de um PLD?
10. Que recurso o fusível de polaridade fornece ao projetista?
11. Qual a principal vantagem de um EPLD?
12. Qual o tipo de dispositivo que proporciona a maior funcionalidade, uma FPGA ou um CPLD?

11-11 RAM SEMICONDUTORA

Lembre-se de que o termo **RAM** significa *memória de acesso aleatório* (*random-access memory*), querendo dizer que qualquer posição de memória possui a mesma facilidade de acesso que qualquer outra. Muitos tipos de memória podem ser classificados como tendo acesso aleatório; entretanto, quando o termo RAM é usado para memórias semicondutoras, estamos nos referindo à memória de escrita e leitura em oposição a ROM. Uma vez que é uma prática comum usar o termo RAM para representar uma memória de escrita e leitura, assim o faremos nas explicações a seguir.

A RAM é usada em computadores para armazenamento *temporário* de programas e dados. Os conteúdos de muitas posições de uma RAM podem ser lidos e escritos à medida que um computador executa um programa. Isso torna necessário que os ciclos de escrita e leitura para RAM sejam mais rápidos para que o desempenho do computador não seja comprometido.

Uma das principais desvantagens de uma RAM é o fato de a mesma ser volátil, isto é, ela vai perder a informação armazenada se a alimentação for desligada ou interrompida. Entretanto, algumas RAMs CMOS consomem tão pouca potência quando estão em modo standby (operações de leitura e escrita não são realizadas) que elas podem ser alimentadas por baterias sempre que a alimentação principal for interrompida. Obviamente, a maior vantagem de uma RAM é o fato de ela poder ser escrita e lida rapidamente com a mesma facilidade.

As explicações a seguir sobre a RAM serão baseadas no que foi explicado para a ROM, uma vez que os conceitos básicos são comuns a ambos os tipos de memória.

11-12 ARQUITETURA DA RAM

Conforme aconteceu com a ROM, é bastante útil pensar numa RAM como um determinado número de registradores onde cada um armazena uma única palavra de dados e possui um único endereço. As RAMs tipicamente são fabricadas com capacidades de 1K, 4K, 8K, 16K, 64K, 128K, 256K e 1.024K, com tamanhos de palavras de 1, 4 ou 8 bits. Como veremos mais tarde, o número de palavras e o tamanho das mesmas podem ser expandidos combinando-se chips de memória.

A Fig. 11-25 mostra a arquitetura simplificada de uma RAM que armazena 64 palavras de quatro bits cada, isto é, uma memória de 64 × 4. Estas palavras têm endereços que vão de 0 a 63_{10}. Para selecionar uma das 64 posições para escrita ou leitura, um endereço binário é fornecido ao circuito decodificador. Uma vez que $64 = 2^6$, o decodificador necessita de seis entradas. Cada endereço ativa uma saída em particular do decodificador, que, por sua vez, habilita o registrador correspondente. Por exemplo, suponha que foi fornecido o endereço

$$A_5A_4A_3A_2A_1A_0 = 011010$$

Uma vez que $011010_2 = 26_{10}$, a saída 26 do decodificador irá para ALTO, selecionando o registrador 26 para escrita ou leitura.

Operação de Leitura

O endereço seleciona o registrador no chip de memória para escrita ou leitura. Para que seja possível *ler* o conteúdo do registrador escolhido, o sinal de entrada de LEITURA/ESCRITA (READ/WRITE — $R/\overline{W}$)* deve estar em 1. Além disso, a entrada de seleção do chip (CHIP SELECT — $\overline{CS}$) deve estar ativa, nesse caso em nível lógico 0. A combinação de

* Alguns fabricantes usam o símbolo $\overline{WE}$ ou $\overline{W}$, em vez de $R/\overline{W}$. Qualquer que seja o caso, o funcionamento é o mesmo.

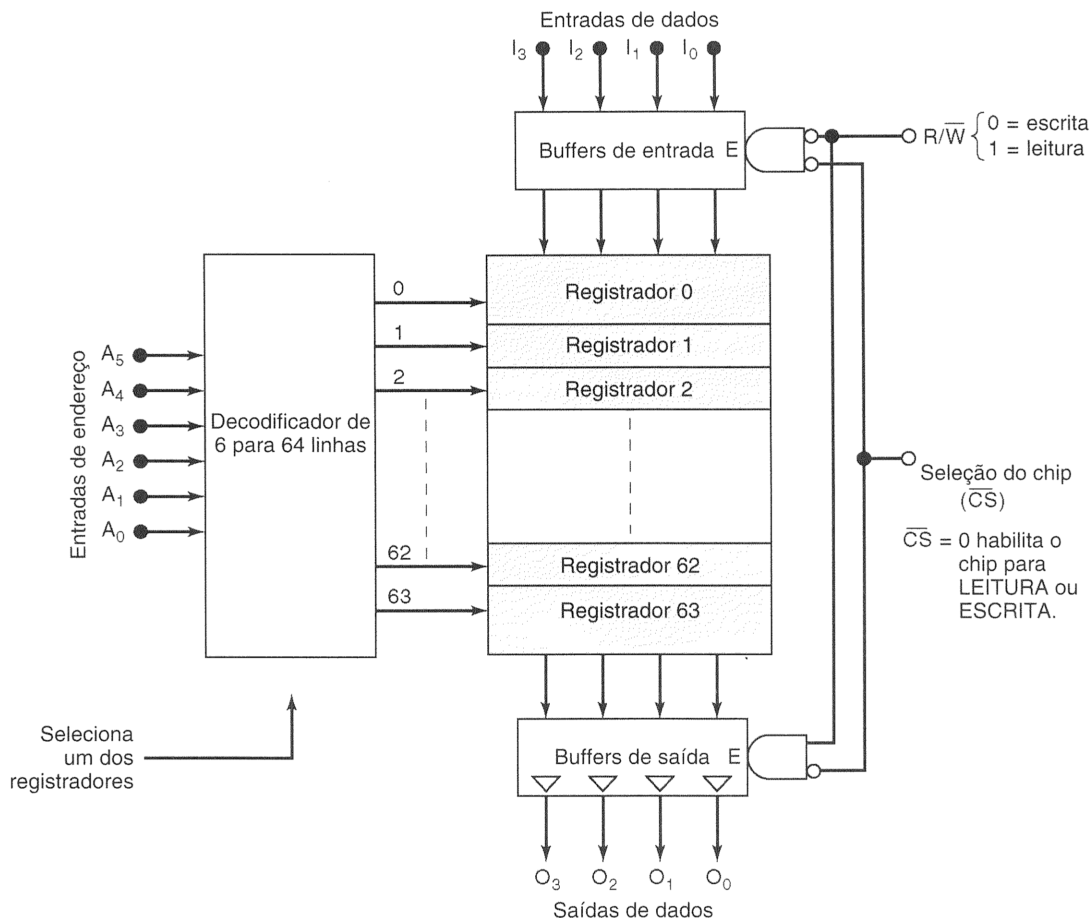


Fig. 11-25 Organização interna de uma RAM de 64 × 4.

$R/\overline{W}$ = 1 e $\overline{CS}$ = 0 habilita os buffers de saída de modo que o conteúdo do registrador aparece nas quatro saídas de dados. A entrada $R/\overline{W}$ = 1 também *inibe* os buffers de entrada, fazendo com que as entradas de dados não afetem a memória durante a operação de leitura.

Operação de Escrita

Para escrever uma palavra de quatro bits no registrador selecionado, é necessário que $R/\overline{W}$ = 0 e $\overline{CS}$ = 0. Essa combinação *habilita* os buffers de entrada para que a palavra de quatro bits fornecida às entradas de dados seja carregada no registrador. A entrada $R/\overline{W}$ em 0 também *inibe* os buffers de saída, que são do tipo tristate, fazendo com que as saídas de dados fiquem em alta impedância durante a operação de escrita. Na operação de escrita, obviamente, o conteúdo que estava anteriormente armazenado é perdido.

Seleção do Chip

A maioria dos chips de memória tem uma ou mais entradas *CS* que são usadas para habilitar ou não o chip. Quando o chip não está habilitado, todas as entradas e saídas de dados estão desabilitadas (alta impedância), de modo que nenhuma operação de leitura ou de escrita pode acontecer. Nesse estado, o conteúdo da memória não é afetado. A razão para a existência de mais de uma entrada de seleção ficará clara quando combinarmos chips de memória para obtermos memórias maiores. Observe que muitos fabricantes chamam estas entradas de HABILITAÇÃO DO CHIP (CHIP ENABLE — *CE*). Quando as entradas *CS* ou *CE* estão no seu estado ativo, diz-se que o chip de memória está *selecionado*; caso contrário, diz-se que ele *não está selecionado*. Muitos CIs de memória são projetados para consumir uma potência muito menor quando eles não estão selecionados. Em grandes sistemas de memória, para uma dada operação de memória, um ou mais chips são selecionados, enquanto outros não são. Falaremos sobre isso mais adiante.

Pinos Comuns de Entrada e Saída

Para diminuir o número de pinos no encapsulamento, os fabricantes geralmente combinam as funções de entrada e saída de dados usando pinos comuns de entrada e saída. A entrada $R/\overline{W}$ controla a função desses pinos de entrada e saída (input/output — I/O). Durante uma operação de leitura, os pinos de I/O funcionam como saídas de dados que reproduzem o conteúdo da posição selecionada. Durante uma operação de escrita, os pinos de I/O funcionam com entradas de dados nas quais os dados a serem escritos devem ser aplicados.

Podemos entender como isso é feito considerando o chip da Fig. 11-25. Com pinos de entrada e saída separados, um total de 18 pinos é necessário (incluindo os pinos de terra e alimentação). Utilizando quatro pinos comuns de I/O, apenas 14 pinos são necessários. Essa economia se torna ainda mais evidente para chips com tamanho de palavra maior.

EXEMPLO 11-9

A RAM NMOS 2147H está organizada como 4K × 1, com entrada e saída de dados separadas e uma entrada de seleção ativa em nível baixo. Desenhe o símbolo lógico para este chip mostrando todas as funções dos pinos.

Solução

O símbolo lógico é mostrado na Fig. 11-26(a).

EXEMPLO 11-10

O MCM6206C é uma RAM CMOS de 32K × 8, pinos de I/O comuns, habilitação do chip ativa em nível BAIXO e habilitação de saída ativa em BAIXO. Desenhe o símbolo lógico.

Solução

O símbolo lógico está mostrado na Fig. 11-26(b).

Questões de Revisão

1. Descreva as condições de entrada necessárias para ler uma palavra de um determinado endereço.
2. Por que alguns chips de RAMs têm pinos de entrada/saída comuns?
3. Quantos pinos são necessários para a RAM MCM6208C de 64K × 4 com uma entrada $\overline{CS}$ e entrada/saída comum?

11-13 RAM ESTÁTICA (SRAM)

A operação da RAM que discutimos até agora se aplica a uma **RAM estática (static RAM — SRAM)**, isto é, aquela que pode armazenar os dados enquanto for mantida a alimentação do chip. Células de uma memória estática são essencialmente flip-flops que permanecem em um determinado estado (armazenando um bit) indefinidamente, desde que a alimentação não seja interrompida. Mais tarde, descreveremos as **RAMs dinâmicas**, que armazenam os dados como cargas em capacitores. No caso das RAMs dinâmicas, o dado armazenado vai desaparecendo gradualmente, devido à descarga do capacitor, sendo, portanto, necessário fazer periodicamente um ***refresh*** dos dados (isto é, recarregar os capacitores).

As RAMs estáticas estão disponíveis nas tecnologias bipolar, MOS e BiCMOS. A maioria das aplicações utiliza RAMs CMOS ou NMOS. Como afirmamos anteriormente, as bipolares são mais rápidas, embora as CMOS venham diminuin-

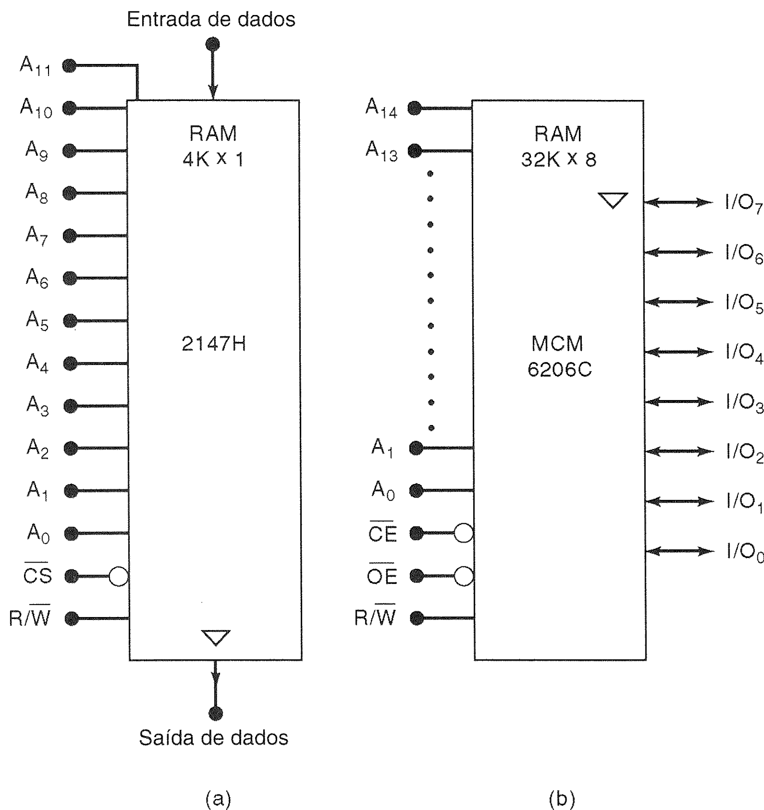


Fig. 11-26 Símbolos lógicos para (a) a RAM 2147H; (b) a RAM MCM6206C.

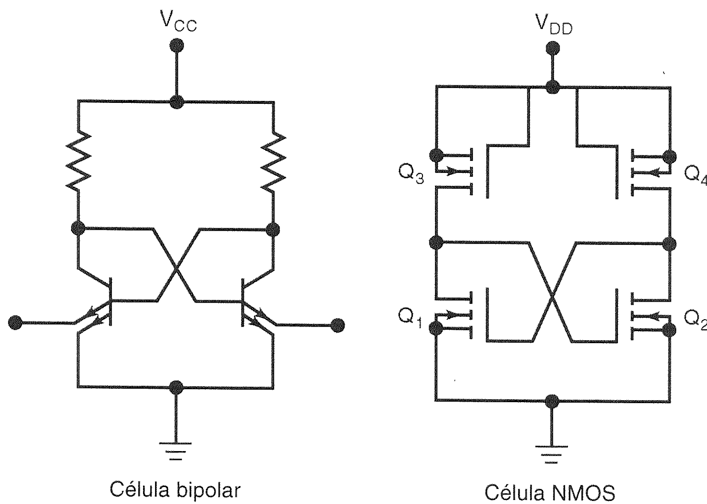


Fig. 11-27 Células típicas de RAMs estáticas bipolar e NMOS.

do bastante essa diferença, e os dispositivos MOS possuem capacidades bem maiores e um menor consumo de potência. A Fig. 11-27 mostra, para fins de comparação, uma célula de memória estática bipolar e uma outra NMOS. A célula bipolar contém dois transistores bipolares e dois resistores, enquanto a NMOS contém quatro MOSFETs canal-N. A célula bipolar necessita de uma área maior do que a célula MOS porque o transistor bipolar é mais complexo do que um MOSFET e também porque a célula bipolar necessita de resistores, enquanto a célula MOS usa MOSFETs como resistores (Q_3 e Q_4). Uma célula CMOS é semelhante a uma NMOS, mas utiliza MOSFETs canal-P no lugar de Q_3 e Q_4. Isso resulta em um menor consumo de potência, mas aumenta a complexidade do chip.

Temporização de uma RAM Estática

Os CIs de RAMs são freqüentemente usados como memória interna de um computador. A CPU (*central processing unit* — unidade central de processamento) realiza continuamente operações de leitura e escrita a uma taxa bastante alta, determinada pelas limitações da CPU. Os chips de memória que estão interfaceados com a CPU devem ser rápidos o suficiente para responderem aos comandos de leitura e escrita da CPU; portanto, o projetista deve se preocupar com as características de temporização das RAMs.

As RAMs têm características de temporização diferentes, mas a maioria delas possui características similares; assim, usaremos um conjunto típico de características para fins de ilustração. A nomenclatura para os diversos parâmetros de temporização varia de um fabricante para outro, mas o significado de cada parâmetro é fácil de descobrir a partir dos diagramas de temporização presentes nas especificações das RAMs. A Fig. 11-28 mostra os diagramas de tempo para ciclos completos de leitura e escrita de uma RAM típica.

Ciclo de Leitura

As formas de onda da Fig. 11-28(a) mostram como as entradas de endereço, de $R/\overline{W}$ e de seleção do chip se comportam durante um ciclo de leitura da memória. Conforme pode ser observado, a CPU fornece esses sinais de entrada para a RAM quando deseja ler os dados de uma posição específica da memória. Embora a RAM possa ter várias entradas de endereço sendo fornecidas pelo barramento de endereços da CPU, por simplicidade o diagrama mostra apenas duas. Os dados de saída da RAM também são mostrados; admitiremos que esta RAM particular tem apenas uma saída de dados. Lembre-se de que os dados de saída da RAM estão ligados ao barramento de dados da CPU (Fig. 11-5).

O ciclo de leitura começa em t_0. Antes desse instante, as entradas de endereço teriam qualquer endereço que estivesse no barramento proveniente da operação anterior. Uma vez que a entrada de seleção do chip não está ativa, ele não responderá ao seu "antigo" endereço. Observe que a linha $R/\overline{W}$ está em ALTO antes de t_0 e permanece nesse estado durante o ciclo de leitura. Na maioria dos sistemas de memória, $R/\overline{W}$ normalmente é mantida em ALTO, exceto durante um ciclo de escrita, quando é colocada em BAIXO. A saída de dados da RAM está em alta impedância, pois $\overline{CS} = 1$.

Em t_0, a CPU aplica um novo endereço às entradas da RAM; este é o endereço da posição que deve ser lida. Após um intervalo de tempo necessário para que as linhas de endereço se estabilizem, a linha $\overline{CS}$ é ativada. A RAM responde a isso colocando os dados da posição endereçada na linha de saída em t_1. O intervalo entre t_0 e t_1 é o tempo de acesso da RAM, t_{ACC}, e é definido como o tempo decorrido entre a aplicação do novo endereço e o aparecimento de dados válidos na saída. O parâmetro t_{CO} é o tempo necessário para que a saída da RAM saia de alta impedância para um nível válido uma vez que $\overline{CS}$ tenha sido ativado.

No instante t_2, $\overline{CS}$ está novamente em ALTO, e a saída da RAM retorna ao estado de alta impedância após um intervalo t_{OD}. Assim, os dados contidos na RAM ficam no barramento de dados entre t_1 e t_3. A CPU pode capturar esses dados do barramento em qualquer ponto desse intervalo. Na maioria dos computadores, a CPU vai utilizar a subida do sinal $\overline{CS}$ em t_2 para capturar esses dados em um dos seus registradores internos.

A duração total do ciclo de leitura, t_{RC}, compreende o intervalo de t_0 a t_4, quando a CPU troca o valor do endereço para o próximo ciclo de leitura ou escrita.

Ciclo de Escrita

A Fig. 11-28(b) mostra o comportamento dos sinais para um ciclo de escrita que se inicia quando a CPU fornece um novo endereço para a RAM no instante t_0. A CPU coloca as linhas $R/\overline{W}$ e $\overline{CS}$ em BAIXO após esperar pelo intervalo de tempo t_{AS}, chamado *tempo de setup do endereço*. Isto dá aos decodificadores da RAM tempo suficiente para responder a esse novo endereço. $R/\overline{W}$ e $\overline{CS}$ são mantidos em BAIXO pelo intervalo de tempo t_W, chamado de tempo de escrita.

Durante o tempo de escrita, no instante t_1, a CPU coloca dados válidos, que devem ser escritos na RAM, no barramento de dados. Estes dados devem ser mantidos na entrada da RAM por um intervalo t_{DS} antes da desabilitação de $R/\overline{W}$ e $\overline{CS}$ em t_2, e por um tempo t_{DH} depois. O intervalo t_{DS} é chamado de *tempo de setup dos dados*, e t_{DH} é chamado de *tempo de hold dos dados*. De modo similar, as

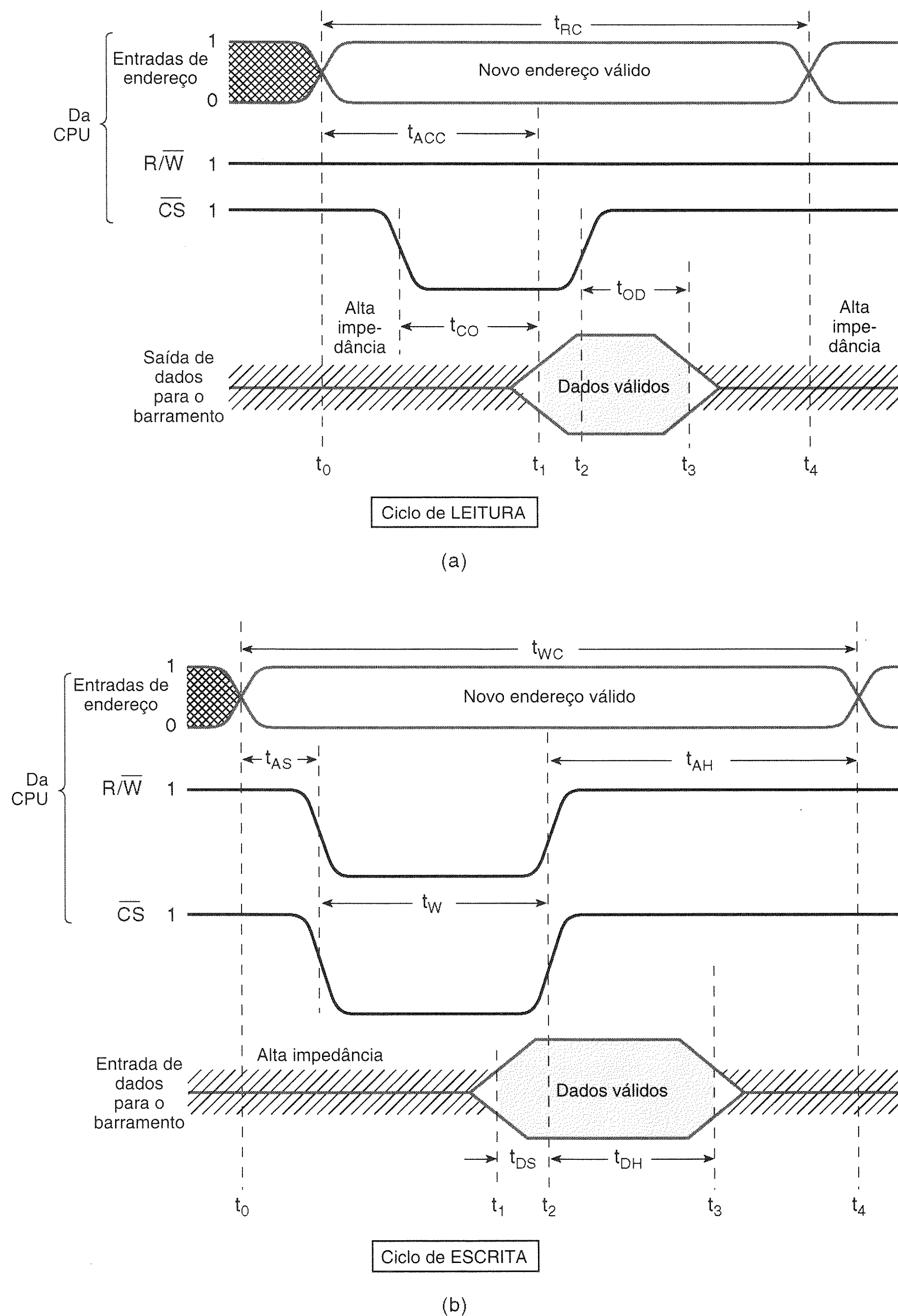


Fig. 11-28 Temporização para uma RAM estática; (a) ciclo de leitura; (b) ciclo de escrita.

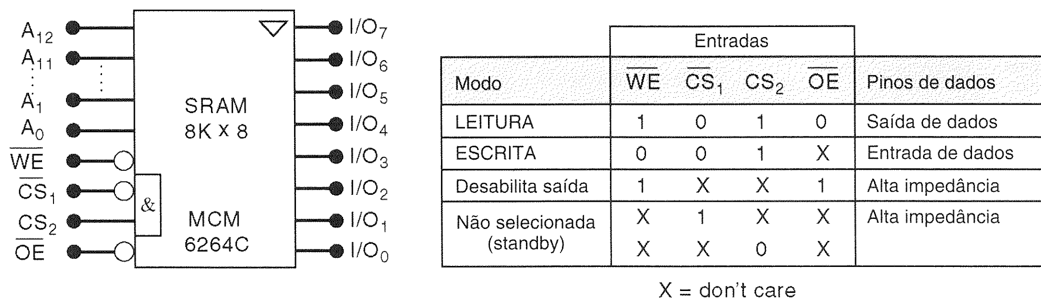
entradas de endereço devem permanecer estáveis durante o tempo de hold do endereço, t_{AH}, após t_2. Se algum desses tempos de setup ou de hold não for respeitado, a operação não será realizada de modo confiável.

O tempo do ciclo de escrita completo, t_{WC}, compreende o intervalo entre t_0 e t_4, quando a CPU troca o valor das linhas de endereço para o próximo ciclo de escrita ou leitura.

O tempo do ciclo de leitura, t_{RC}, e o tempo do ciclo de escrita, t_{WC}, determinam quão rápido uma memória pode operar. Por exemplo, em uma aplicação real, uma CPU vai ler palavras de dados sucessivas, uma após a outra. Se a memória possui um t_{RC} de 50 ns, a CPU pode ler uma palavra a cada 50 ns, ou 20 milhões de palavras por segundo; com um t_{RC} = 10 ns, a CPU pode ler 100 milhões de palavras por segundo. A Tabela 11-3 mostra os tempos míni-

TABELA 11-3

Dispositivo	t_{RC} (min) (ns)	t_{WC} (min) (ns)
CMOS MCM6206C, 32K × 8	15	15
NMOS 2147H, 4K × 1	35	35
BiCMOS MCM6708A, 64K × 4	8	8



	Entradas				
Modo	$\overline{WE}$	$\overline{CS}_1$	CS_2	$\overline{OE}$	Pinos de dados
LEITURA	1	0	1	0	Saída de dados
ESCRITA	0	0	1	X	Entrada de dados
Desabilita saída	1	X	X	1	Alta impedância
Não selecionada (standby)	X X	1 X	X 0	X X	Alta impedância

X = don't care

Fig. 11-29 Símbolo lógico e tabela de funcionamento para a RAM CMOS MCM6264C.

mos para os ciclos de leitura e escrita de algumas SRAMs representativas.

Um Chip de Memória Estática Real

Um exemplo de CI de memória estática é a RAM CMOS de 8K × 8 MCM6264C com tempos de ciclos de leitura e escrita de 12 ns e consumo de potência em standby de apenas 100 mW. O símbolo lógico para este CI pode ser visto na Fig. 11-29. Observe que ele possui 13 entradas de endereço, uma vez que $2^{13} = 8.192 = 8K$, e oito linhas de dados bidirecionais. As quatro entradas de controle determinam o modo de operação de acordo com a tabela na Fig. 11-29.

A entrada $\overline{WE}$ tem o mesmo significado que a entrada $R/\overline{W}$ que vínhamos utilizando. Um nível BAIXO em $\overline{WE}$ vai escrever dados na RAM, desde que o dispositivo esteja selecionado, isto é, ambas as entradas de seleção estejam ativas. Observe que o símbolo "&" é usado para indicar que *ambas* devem estar ativas. Um nível ALTO em $\overline{WE}$ vai produzir uma operação de leitura, desde que o dispositivo esteja selecionado e que os buffers de saída estejam habilitados por $\overline{OE}$ = BAIXO. Quando não está selecionado, o dispositivo está no modo de baixo consumo, e nenhuma das outras entradas tem efeito.

A maioria dos dispositivos que estudamos neste capítulo está disponível de um grande número de fabricantes. Cada fabricante pode oferecer um determinado número de dispositivos de mesma capacidade mas com diferentes características e recursos. Além disso, existem vários tipos de encapsulamento disponíveis, tais como DIP, PLCC e diversos outros, para montagem em superfície.

Observando atentamente os dispositivos de memória que temos estudado, você irá notar diversas semelhanças. Por exemplo, olhe os chips de memória na Fig. 11-30 e anote a sua numeração. O fato de que a mesma função é associada aos mesmos pinos em todos os dispositivos fabricados por diversas companhias não é coincidência. Os padrões industriais criados pelo **JEDEC** (Joint Electronic Device Engineering Council) fizeram com que os dispositivos de memória sejam bastante intercambiáveis.

EXEMPLO 11-11

Um sistema contém uma ROM de 8K × 8 (2764) e duas RAMs também de 8K × 8 (6264). A capacidade total da ROM está sendo usada para armazenar um programa para o microprocessador. Queremos dotar o sistema de algum tipo

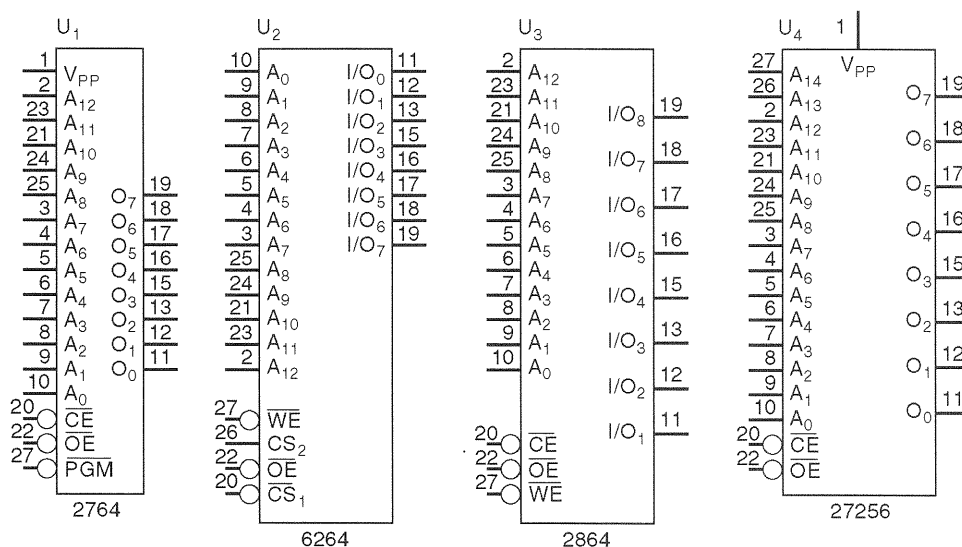


Fig. 11-30 Padrão JEDEC para encapsulamento de memória.

de armazenamento não-volátil de leitura e escrita. O circuito atual pode ser modificado para acomodar essa nova revisão?

Solução

A EEPROM 2864 pode ser colocada diretamente no lugar de uma das RAMs. A única diferença funcional é que a EEPROM possui um ciclo de escrita muito mais longo. Essa diferença geralmente é tratada alterando-se a parte do programa do microprocessador que está utilizando essa memória. Uma vez que não existe espaço disponível para essas mudanças, precisaremos de uma ROM maior. A 27C256 tem basicamente a mesma pinagem que a 2764 e possui 32K × 8. Agora, simplesmente, precisamos conectar duas linhas de endereço a mais (A_{13} e A_{14}) ao soquete da ROM e trocar o chip antigo (2764) pelo novo (27C256).

Muitos sistemas de memória tiram proveito da versatilidade que o padrão JEDEC proporciona. Os pinos que são comuns a todos os dispositivos são conectados permanentemente aos barramentos do sistema. Os poucos pinos que possuem funções diferentes nos vários dispositivos são conectados a um circuito que permite facilmente modificar a configuração do sistema para o tamanho e o tipo de memória desejados. Isto permite que o usuário reconfigure o hardware sem a necessidade de cortar trilhas ou fazer soldas na placa. Esse circuito de configuração pode ser tão simples quanto estrapes removíveis ou DIP switches que o usuário configura, ou tão complicado quanto um dispositivo de lógica programável que o computador pode configurar ou modificar para atender às necessidades do sistema.

Questões de Revisão

1. Como uma célula de memória estática difere de uma de memória dinâmica?
2. Que tecnologia de memória geralmente consome menos potência?
3. Que dispositivo coloca dados no barramento de dados durante o ciclo de leitura?
4. Que dispositivo coloca dados no barramento de dados durante o ciclo de escrita?
5. Quais os parâmetros de temporização da RAM que determinam a sua velocidade de operação?
6. *Verdadeiro ou falso*: Um nível BAIXO em $\overline{OE}$ habilita os buffers de saída na MCM6264C, desde que ambas as entradas de seleção estejam ativas.
7. O que deve ser feito com o pino 26 e o pino 27 se uma 27256 for substituída por uma 2764?

11-14 RAM DINÂMICA (DRAM)

RAMs dinâmicas são fabricadas usando tecnologia MOS e se destacam por sua alta capacidade, seu baixo consumo de potência e sua velocidade de operação moderada. Conforme afirmamos anteriormente, ao contrário das RAMs estáticas, que armazenam informações em flip-flops, as RAMs dinâmicas armazenam 1s e 0s como cargas em um pequeno capacitor MOS (geralmente alguns picofarads). Como existe uma tendência para a fuga de cargas após um período de tempo, as RAMs dinâmicas necessitam de uma recarga periódica de suas células de memória. Isto é chamado de **refresh** da memória dinâmica. Nas DRAMs mais modernas, cada célula de memória deve ser recarregada, geralmente, a cada 2, 4, ou 8 ms, ou os dados serão perdidos.

A necessidade de refresh é uma desvantagem da RAM dinâmica quando comparada com a RAM estática porque ela pode necessitar de circuitos externos de suporte. Algumas DRAMs possuem circuitos de controle de refresh incorporados e, portanto, não necessitam de hardware externo extra; entretanto, existe a necessidade de que se obedeça à temporização especial das entradas de controle do chip. Além disso, como veremos, as entradas de endereço de uma DRAM necessitam de um tratamento mais complexo do que as da SRAM. Portanto, somando-se tudo isso, é muito mais difícil projetar um sistema com DRAM do que com SRAM. Entretanto, sua capacidade de armazenamento bem maior e seu consumo de potência bem menor tornam a DRAM a escolha certa quando as considerações principais no projeto são manter o custo, o tamanho e o consumo de potência pequenos.

Para aplicações em que a velocidade e a complexidade reduzida são mais importantes do que o custo, o tamanho e a potência, as RAMs estáticas são a melhor opção. Elas são geralmente mais rápidas do que as RAMs dinâmicas e não necessitam de refresh. Elas são mais fáceis de se utilizar em um projeto, mas não podem competir com a capacidade de armazenamento mais alta e o baixo consumo das RAMs dinâmicas.

Devido à simplicidade da estrutura de sua célula, as DRAMs são geralmente quatro vezes mais densas do que as SRAMs. Essa grande densidade permite que quatro vezes mais capacidade de memória seja colocada em uma única placa, ou também pode-se dizer que elas necessitam de um quarto do espaço em placa para a mesma quantidade de memória. O custo por bit de armazenamento com memória dinâmica é geralmente um quinto, até um quarto, do custo para o caso de RAMs estáticas. Existe também uma economia gerada pelo baixo consumo da RAM dinâmica. Tipicamente, uma RAM dinâmica consome cerca de um sexto a metade do que consumiria uma RAM estática. Isso permite o uso de fontes de alimentação menores e mais baratas.

As principais aplicações da RAM estática ocorrem em áreas onde apenas pequenas quantidades de memória são necessárias (até 64K) ou quando alta velocidade é requisitada. Muitos equipamentos controlados por microprocessadores têm requisitos de memória muito pequenos. Alguns instrumentos como osciloscópios com memória digital e analisadores lógicos necessitam de memórias muito rápidas. Para aplicações como essas, as SRAMs são normalmente utilizadas.

A memória interna principal da maioria dos computadores pessoais (por exemplo: Macs ou PCs rodando Windows) usa DRAMs por causa de sua alta capacidade e baixo consumo. Esses computadores, entretanto, às vezes utilizam pequenas quantidades de memória estática para funções que necessitam de grande velocidade, tais como tabelas, memória de vídeo e memória cache.

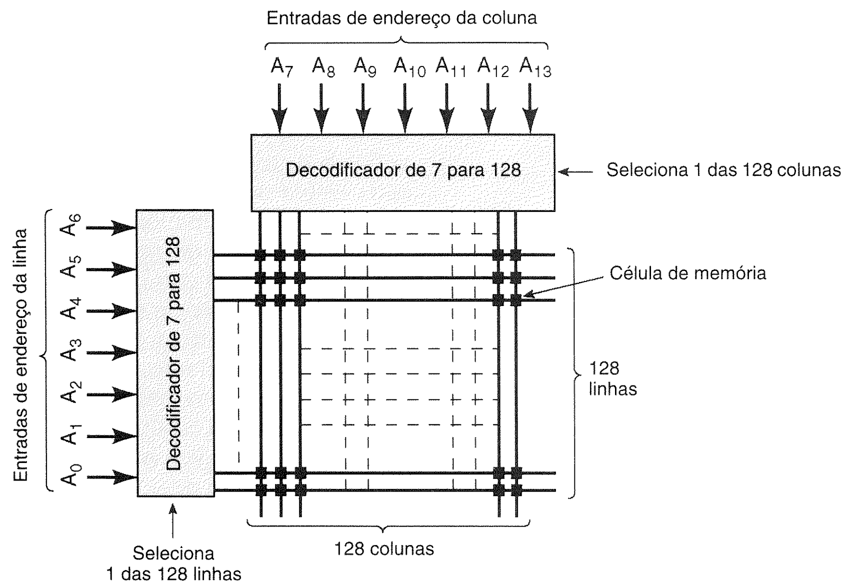


Fig. 11-31 Organização das células em uma memória dinâmica de 16K × 1.

> **Questões de Revisão**
>
> 1. Quais são as principais desvantagens da RAM dinâmica quando comparada com a estática?
> 2. Relacione as vantagens da RAM dinâmica quando comparada com a estática.
> 3. Que tipo de memória você esperaria que fosse utilizada em equipamentos alimentados por bateria?

11-15 ESTRUTURA E OPERAÇÃO DA RAM DINÂMICA

A arquitetura interna de uma RAM dinâmica pode ser visualizada como uma matriz de células, conforme está ilustrado na Fig. 11-31. Nesse caso, as 16.384 células estão organizadas em uma matriz de 128 × 128. Cada célula ocupa uma posição correspondente a uma única linha e uma única coluna na matriz. Quatorze entradas de endereço são necessárias para selecionar uma das células (2^{14} = 16.384); os bits de endereço mais baixos, A_0 a A_6, selecionam a linha, e os bits de mais alta ordem, A_7 a A_{13}, selecionam a coluna. Cada endereço de 14 bits seleciona uma única célula para leitura ou escrita. A estrutura mostrada na Fig. 11-31 é de um chip de memória dinâmica de 16K × 1. Os chips de DRAMs estão disponíveis atualmente com capacidades de até 64 Mbits em várias configurações. As DRAMs com tamanho de palavra igual a quatro bits (ou maior) têm a organização das células semelhante àquela vista na Fig. 11-31, exceto que cada posição na matriz contém quatro células, e cada endereço fornecido seleciona um grupo de quatro células para uma operação de leitura ou escrita. Como veremos mais adiante, tamanhos de palavra maiores também podem ser obtidos combinando-se diversos chips em arranjos apropriados.

A Fig. 11-32 é uma representação simbólica de uma célula de memória dinâmica e de seus circuitos associados. Muitos detalhes desse circuito não são mostrados, mas esse diagrama simplificado pode ser usado para descrever as idéias principais que estão envolvidas na leitura e escrita de uma DRAM. As chaves SW1 a SW4 são na verdade MOSFETs controlados pelas várias saídas do decodificador de endereços e pelo sinal $R/\overline{W}$. O capacitor, obviamente, é a célula de armazenamento.

Para escrever em uma célula, os sinais provenientes da decodificação de endereços e da lógica de leitura/escrita fecham as chaves SW1 e SW2, mantendo SW3 e SW4 abertas. Isto conecta a entrada de dados a *C*. Um nível lógico 1

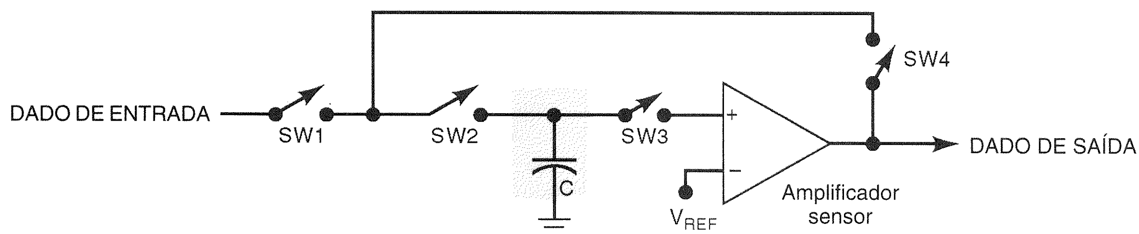


Fig. 11-32 Representação simbólica de uma célula de memória dinâmica. Durante a operação de escrita, as chaves semicondutoras SW1 e SW2 estão fechadas. Durante a operação de leitura, todas as chaves estão fechadas, exceto SW1.

na entrada de dados carrega C, e um nível 0 o descarrega. Então, depois disso, as chaves são abertas, de modo que C fica desconectado do resto do circuito. Idealmente, C iria manter sua carga indefinidamente, mas existe sempre um caminho para a fuga de corrente através das chaves desligadas, de modo que C vai sendo descarregado gradualmente.

Para ler uma célula, as chaves SW2, SW3 e SW4 são fechadas, e SW1 permanece aberta. Isso conecta a tensão armazenada no capacitor ao *amplificador sensor*. Este amplificador compara a tensão do capacitor com um valor de referência para determinar se esta corresponde a um nível 0 ou 1, e fornece 0 V ou 5 V bem-definido para a saída de dados. Esta saída também está conectada a C (SW2 e SW4 estão fechadas) e restaura a tensão do capacitor através de carga ou descarga. Em outras palavras, o bit da célula de memória é restaurado a cada leitura.

Multiplexação de Endereços

A DRAM de 16K × 1 que aparece na Fig. 11-31 está quase obsoleta. Apenas alguns poucos fabricantes ainda fabricam DRAMs de capacidade tão baixa. Ela possui 14 entradas de endereço; uma DRAM de 64K × 1 teria 16 entradas de endereço. Uma DRAM de 1M × 1 precisa de 20 entradas de endereço, e uma DRAM de 4M × 1 precisa de 22 entradas de endereço. DRAMs de alta capacidade como essas necessitariam de muitos pinos se cada entrada de endereço correspondesse a um pino do chip somente para ela. De modo a reduzir o número de pinos em chips DRAM de alta capacidade, os fabricantes utilizam a técnica de **multiplexação de endereços**, em que cada pino de entrada de endereço pode acomodar dois bits de endereço diferentes. Essa economia no número de pinos se traduz em uma significativa diminuição do tamanho do encapsulamento dos CIs. Isso é muito importante em placas de memória de alta capacidade, em que você deseja maximizar a quantidade de memória que pode ser colocada em uma placa.

Nas explicações a seguir, descrevemos a ordem na qual o endereço é multiplexado no interior dos chips DRAM. Devemos notar que nas DRAMs mais antigas, de menor capacidade, existia uma convenção na qual se apresentavam primeiro os bits de endereço de mais baixa ordem para especificar a linha e depois os bits de mais alta ordem para especificar a coluna. As DRAMs mais novas e os controladores que realizam a multiplexação agora utilizam uma convenção oposta, isto é, apresentam-se os bits de ordem mais alta como o endereço da linha e depois os bits de ordem mais baixa como o endereço da coluna. Faremos a descrição dessa convenção mais recente, mas você deve estar ciente da outra convenção, principalmente quando estudar sistemas mais antigos.

Usaremos a DRAM TMS44100 de 4M × 1 da Texas Instruments para ilustrar a operação de chips DRAM atuais. O diagrama de blocos funcionais da estrutura interna do chip (mostrada na Fig. 11-33) é bem semelhante àqueles que você encontrará nos manuais. A disposição da matriz de memória nesse diagrama pode parecer um pouco complicada à primeira vista, mas podemos pensar que ela é apenas uma versão maior da DRAM de 16K × 1 da Fig. 11-31. Funcionalmente, ela é uma matriz de células organizadas em 2.048 linhas por 2.048 colunas. Uma única coluna é selecionada pelo circuito decodificador de endereços, que pode ser visto como um decodificador de 11 para 2.048. Analogamente, uma única coluna é selecionada por um circuito que é efetivamente um decodificador de 11 para 2.048. Como as linhas de endereço estão multiplexadas, as 22 linhas de endereço não são apresentadas simultaneamente. Observe que existem apenas 11 linhas de endereço e que elas vão para o registrador do endereço da linha e para o registrador do endereço da coluna. Cada um dos registradores armazena metade dos 22 bits de endereço. O registrador da linha armazena a metade superior, e o registrador da coluna armazena a metade inferior. Duas entradas muito importantes controlam quando o endereço deve ser armazenado no registrador. O **strobe do endereço da linha (*row address strobe* — $\overline{RAS}$)** é o clock do registrador do endereço da linha. O **strobe do endereço da coluna (*column address strobe* — $\overline{CAS}$)** é o clock do registrador do endereço da coluna.

Os 22 bits de endereços são apresentados a essa DRAM em duas etapas utilizando $\overline{RAS}$ e $\overline{CAS}$. A temporização desse processo pode ser vista na Fig. 11-33(b). Inicialmente, $\overline{RAS}$ e $\overline{CAS}$ estão ambos em ALTO. No instante t_0, os 11 bits de endereço da linha (A_{11} a A_{21}) são aplicados nas entradas de endereço. Após aguardar o tempo necessário para satisfazer o tempo de setup (t_{RS}) do registrador do endereço da linha, a entrada $\overline{RAS}$ é colocada em BAIXO em t_1. Essa transição negativa carrega o endereço da linha no registrador da linha, fazendo com que A_{11} a A_{21} apareçam nas entradas do decodificador de linha. O nível BAIXO em $\overline{RAS}$ também habilita esse decodificador, de modo que ele possa decodificar o endereço da linha e selecionar uma das linhas da matriz.

No instante t_2, os 11 bits de endereço da coluna (A_0 a A_{10}) são aplicados nas entradas de endereço. Em t_3, a entrada $\overline{CAS}$ é colocada em nível BAIXO para que o endereço da coluna seja carregado em seu registrador. A entrada $\overline{CAS}$ em BAIXO também habilita o decodificador da coluna para que ele possa decodificar o endereço e selecionar uma coluna da matriz.

Nesse ponto, as duas partes do endereço estão em seus respectivos registradores, os decodificadores já os decodificaram e selecionaram uma célula correspondente ao endereço da linha e da coluna, então uma operação de leitura ou escrita pode ser realizada nessa célula, da mesma maneira que seria realizada em uma RAM estática.

Você deve ter notado que essa DRAM não tem uma entrada de seleção de chip (CS). Os sinais $\overline{RAS}$ e $\overline{CAS}$ realizam essa função de seleção, uma vez que ambos devem estar em nível BAIXO para que os decodificadores selecionem uma célula para escrita ou leitura.

EXEMPLO 11-12

Quantos pinos são economizados usando-se a multiplexação de endereços em uma DRAM de 16M × 1?

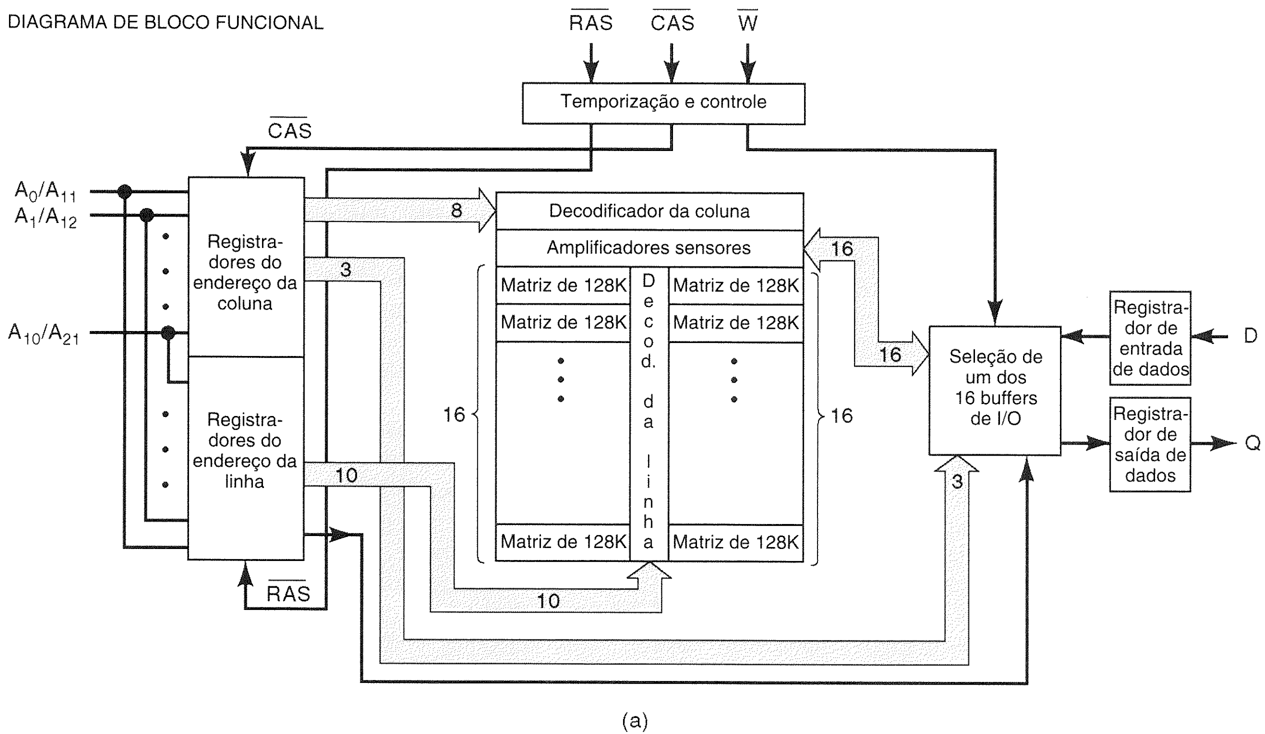
Solução

Doze entradas de endereço são usadas em vez de 24. As entradas $\overline{RAS}$ e $\overline{CAS}$ são adicionadas, e não há necessidade de $\overline{CS}$. Portanto, a economia líquida é de *onze* pinos.

Em um computador típico, as entradas de endereço para a memória vêm da unidade central de processamento (CPU). Quando a CPU deseja acessar uma posição de memória particular, ela gera o endereço completo e o coloca nas linhas de endereço, que, juntas, constituem o **barramento de endereço**. Isto é mostrado na Fig. 11-34(a) para uma pequena memória de 64K palavras, e que portanto necessita de 16 linhas de endereço que vão diretamente da CPU para a memória.

Esse arranjo funciona para uma ROM ou para uma RAM estática, mas ele deve ser modificado para DRAMs que usam a multiplexação de endereços. No caso de todos os 64K da memória serem DRAM, ela necessita de apenas 8 entradas de endereço. Isso significa que as 16 linhas de endereço, que vêm do barramento de endereço da CPU, devem ser aplicadas em um multiplexador que irá transmitir 8 bits de endereço de cada vez para as entradas de endereço da memória. Isso está mostrado simbolicamente na Fig. 11-34(b). A entrada de seleção do multiplexador, chamada de *MUX*, controla se as linhas de endereço A_0 a A_7 ou as linhas de endereço A_8 a A_{15} estarão presentes nas entradas da DRAM.

A temporização do sinal MUX deve estar sincronizada com os sinais $\overline{RAS}$ e $\overline{CAS}$, que são responsáveis por colocar os endereços nos registradores da DRAM. Isso pode ser visto na Fig. 11-35. O sinal *MUX* deve estar em BAIXO quando o $\overline{RAS}$ for pulsado para BAIXO, de modo que as linhas de endereço A_8 a A_{15} da CPU chegarão até as entradas de en-



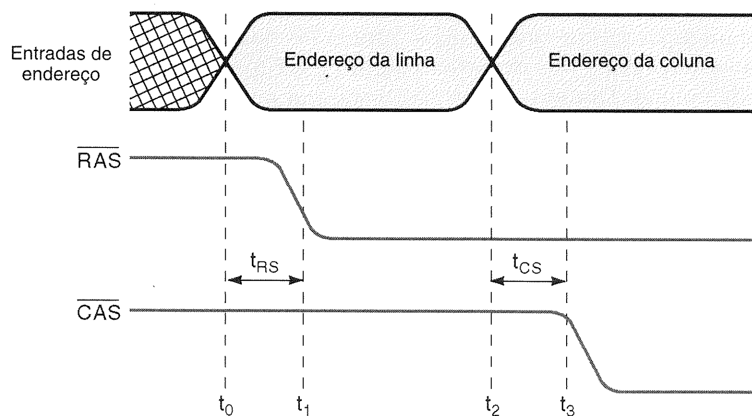
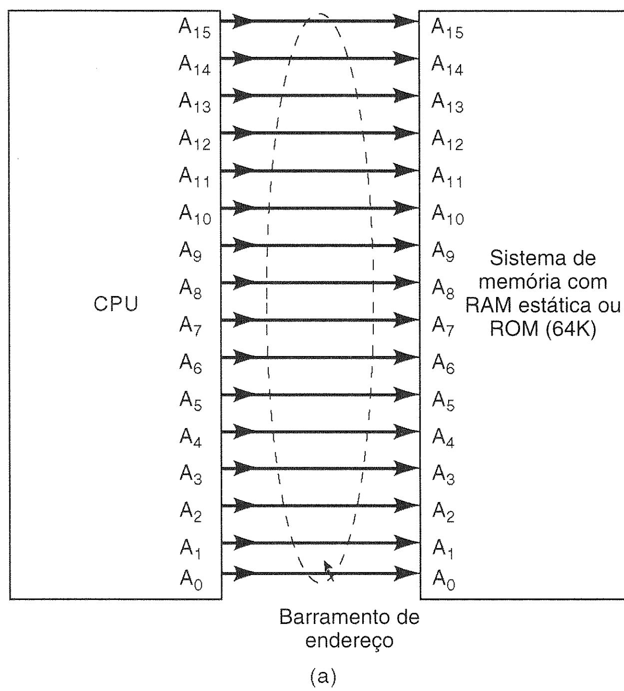


Fig. 11-33 (a) Arquitetura simplificada da TMS44100, DRAM de 4M × 1; (b) temporização de $\overline{RAS}$ e $\overline{CAS}$. (Cortesia: Texas Instruments.)



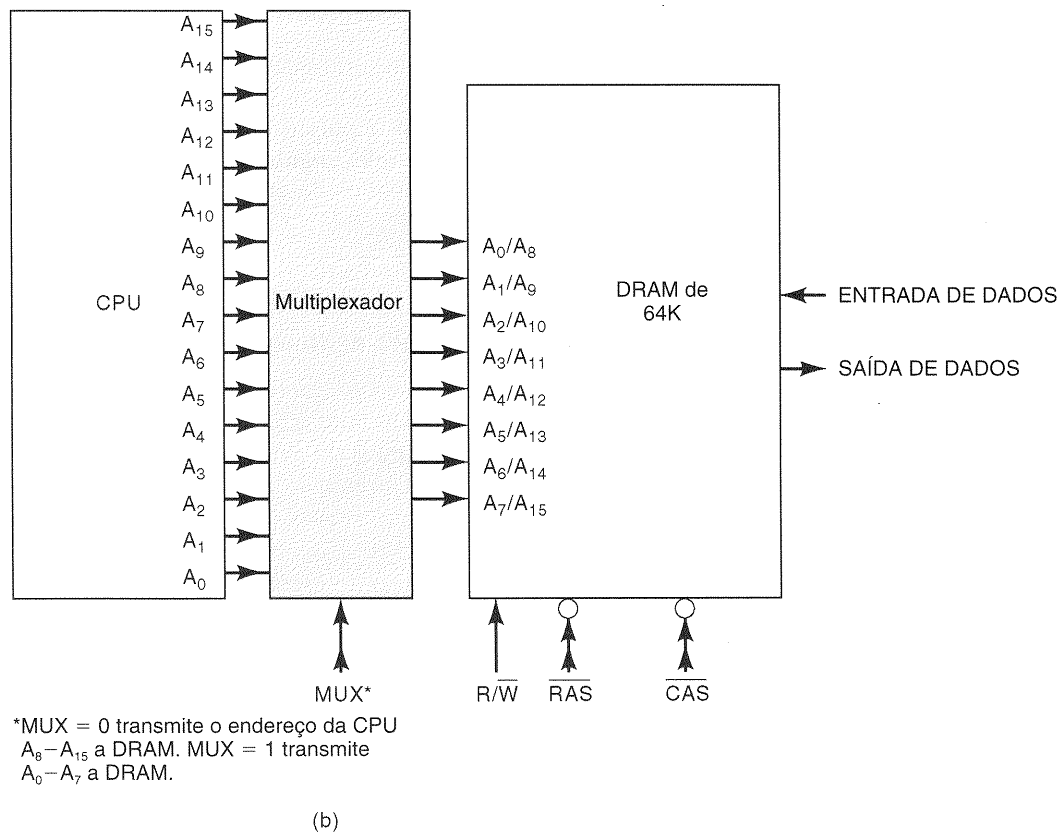


Fig. 11-34 (a) Barramento de endereço da CPU conectado numa ROM ou numa memória estática; (b) linhas de endereço da CPU conectadas em um multiplexador que é usado para multiplexar essas linhas para a DRAM.

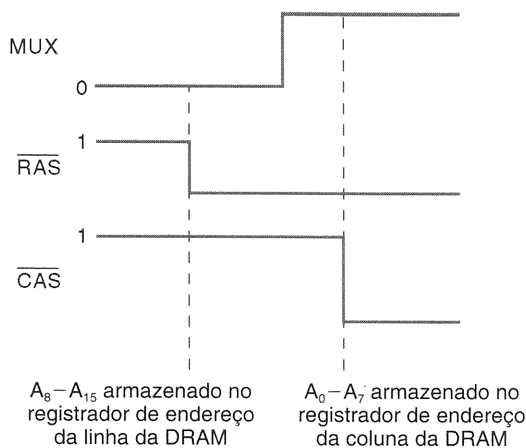


Fig. 11-35 Temporização que deve ser obedecida para a multiplexação do endereço.

dereço da DRAM, sendo que a parte do endereço presente nessas linhas será carregada na descida de $\overline{RAS}$. Do mesmo modo, *MUX* deve estar em ALTO quando $\overline{CAS}$ for pulsado para BAIXO, de modo que A_0 e A_7, provenientes da CPU, estarão presentes nas entradas da DRAM e serão carregados na descida de $\overline{CAS}$.

O circuito de temporização e multiplexação real não está mostrado aqui, e será visto nos problemas de final de capítulo (Problemas 11-26 e 11-27).

Questões de Revisão

1. Descreva a estrutura da matriz de uma DRAM de 64K × 1.
2. Qual é a vantagem da multiplexação de endereços?
3. Quantas entradas de endereço teria uma DRAM de 1M × 1?
4. Quais são as funções dos sinais $\overline{RAS}$ e $\overline{CAS}$?
5. Qual é a função do sinal *MUX*?

11-16 CICLOS DE LEITURA/ESCRITA DA RAM DINÂMICA

A temporização das operações de leitura e escrita em uma DRAM é muito mais complexa do que em uma RAM estática, existindo diversos requisitos críticos de temporização que o projetista deve levar em conta. Neste momento, uma discussão detalhada desses requisitos provavelmente iria apenas causar confusão em vez de ajudar você a compreender o funcionamento de uma DRAM. Vamos nos concentrar na seqüência de temporização básica para as operações de leitura e escrita para um pequeno sistema com memória DRAM como o da Fig. 11-34(b).

Ciclo de Leitura de uma DRAM

A Fig. 11-36 mostra o comportamento típico dos sinais durante uma operação de leitura. Consideramos que o sinal de $R/\overline{W}$ está em ALTO durante toda a operação. A seguir, faremos uma descrição passo a passo dos eventos que ocorrem nos instantes mostrados no diagrama.

- t_0: *MUX* é colocado em BAIXO para aplicar os bits de endereço da linha (A_8 a A_{15}) nas entradas de endereço da DRAM.
- t_1: $\overline{RAS}$ é colocado em BAIXO para carregar o endereço da linha na DRAM.
- t_2: *MUX* vai para ALTO para colocar o endereço da coluna (A_0 a A_7) nas entradas de endereço da DRAM.
- t_3: $\overline{CAS}$ vai para BAIXO para carregar o endereço da coluna na DRAM.
- t_4: A DRAM responde colocando o conteúdo da célula de memória selecionada na linha DATA OUT.
- t_5: *MUX*, $\overline{RAS}$, $\overline{CAS}$ e DATA OUT retornam aos seus estados iniciais.

Ciclo de Escrita de uma DRAM

A Fig. 11-37 mostra o comportamento típico dos sinais durante uma operação de escrita na DRAM.

- t_0: Um nível BAIXO em *MUX* coloca o endereço da linha nas entradas da DRAM.
- t_1: A descida de $\overline{RAS}$ carrega o endereço da linha na DRAM.
- t_2: *MUX* vai para ALTO para colocar o endereço da coluna nas entradas de endereço da DRAM.
- t_3: A descida de $\overline{CAS}$ carrega o endereço da coluna na DRAM.
- t_4: Dados que serão escritos são colocados na linha DATA IN.
- t_5: $R/\overline{W}$ é pulsado em BAIXO para escrever os dados na célula selecionada.
- t_6: Os dados de entrada são removidos de DATA IN.
- t_7: *MUX*, $\overline{RAS}$, $\overline{CAS}$ e $R/\overline{W}$ retornam aos seus estados iniciais.

Questões de Revisão

1. *Verdadeiro ou falso*:
   - **(a)** Durante o ciclo de leitura, o sinal $\overline{RAS}$ é ativado antes do sinal $\overline{CAS}$.
   - **(b)** Durante a operação de escrita, o sinal $\overline{CAS}$ é ativado antes do sinal $\overline{RAS}$.
   - **(c)** $R/\overline{W}$ é mantido em BAIXO durante toda a operação de escrita.
   - **(d)** As entradas de endereço de uma DRAM mudam duas vezes durante uma operação de leitura ou escrita.
2. Que sinal garante que a parte correta do endereço completo aparece nas entradas da DRAM?

11-17 REFRESH DA RAM DINÂMICA

Uma célula de RAM dinâmica é refrescada toda vez que uma operação de leitura é realizada nesta célula. Cada célula de memória deve ser refrescada periodicamente (geralmente

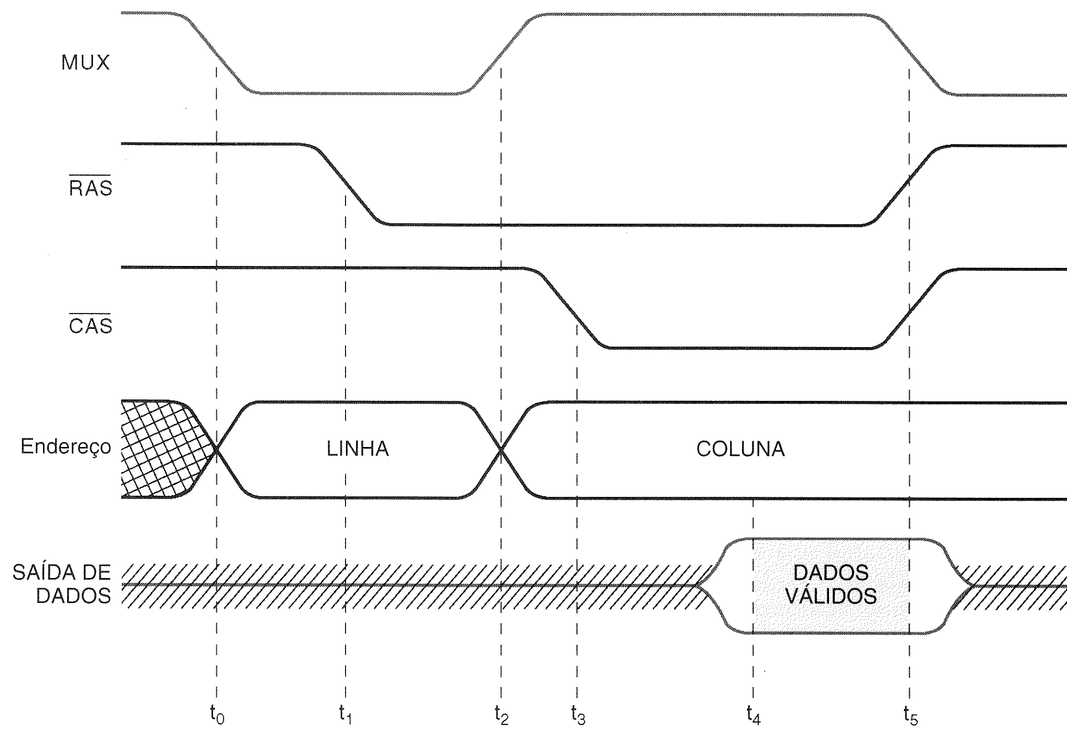


Fig. 11-36 Comportamento dos sinais durante uma operação de leitura em uma RAM dinâmica. A entrada $R/\overline{W}$ (não mostrada) fica em ALTO durante toda a operação.

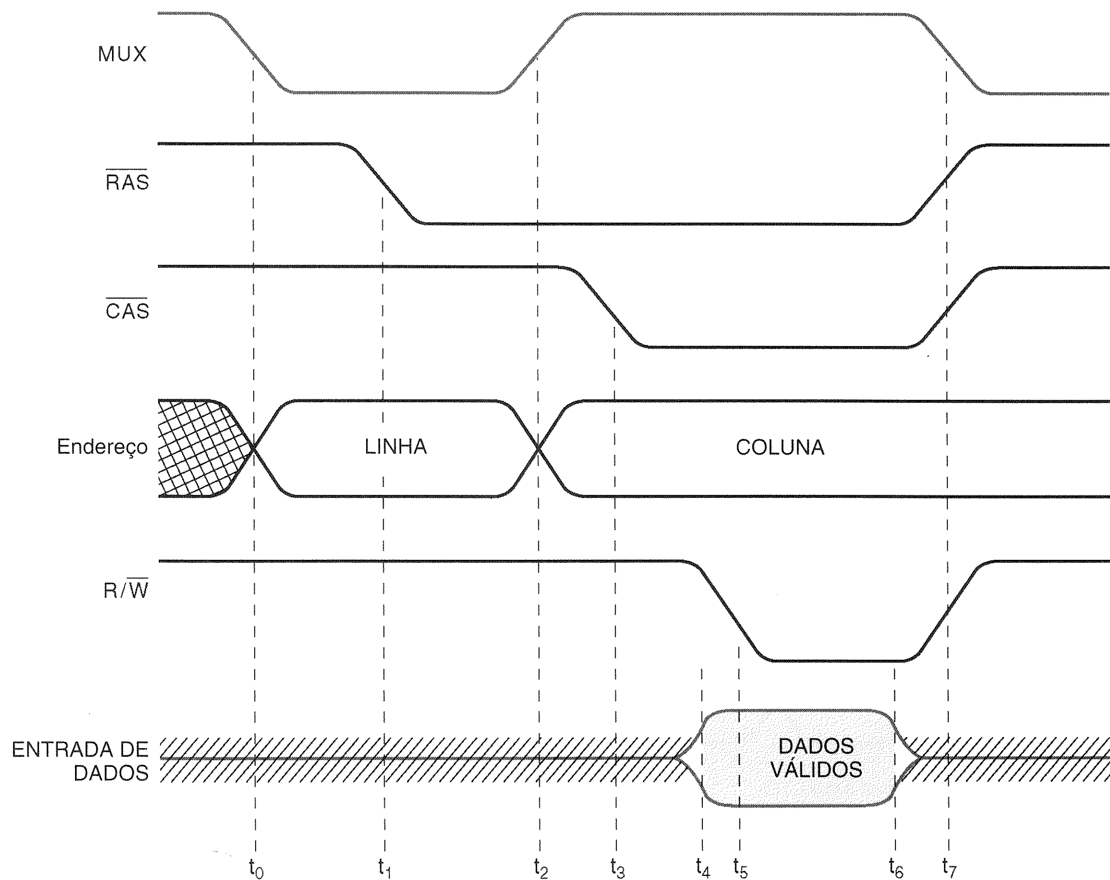


Fig. 11-37 Comportamento dos sinais durante uma operação de escrita em uma RAM dinâmica.

de 4 a 16 ms, dependendo do dispositivo), pois do contrário seu dado será perdido. Este requisito poderia parecer extremamente difícil, ou mesmo impossível, de atender no caso de DRAMs de grande capacidade. Por exemplo, uma DRAM de 1M × 1 possui 10^{20} = 1.048.576 células. Para garantir que cada célula fosse refrescada a cada 4 ms, seria necessário que as operações de leitura fossem realizadas em endereços sucessivos a uma taxa correspondente a uma operação a cada 4 ns (4 ms/1.048.576 ≈ 4 ns). Isto é excessivamente rápido para qualquer chip DRAM. Felizmente, os fabricantes projetaram os chips de DRAM de modo que

sempre que uma operação de leitura é realizada numa célula, todas as células naquela linha são refrescadas.

Assim, é necessário fazer apenas uma operação de leitura em cada *linha* de um arranjo DRAM a cada 4 ms para garantir que cada *célula* do arranjo seja refrescada. Em relação à DRAM de 4M × 1 da Fig. 11-33(a), se qualquer endereço fosse colocado no registrador de endereço da linha, todas as 2.048 células naquela linha seriam automaticamente refrescadas.

Obviamente, essa característica especial de refresh por linha torna mais fácil manter todas as células da DRAM refrescadas. Entretanto, durante a operação normal do sistema na qual a DRAM está funcionando, não é provável que uma operação de leitura seja realizada em cada linha da DRAM dentro do limite de tempo previsto para o refresh. Portanto, algum tipo de lógica de controle de refresh é necessário, ou externamente ao chip de DRAM ou como parte de seus circuitos internos. Em ambos os casos, existem dois modos de refrescamento: o refresh *em rajada* (*burst*) e o refresh *distribuído*.

No modo de refresh em rajada, a operação normal da memória é suspensa, e cada linha da DRAM é refrescada em seqüência até que todas as linhas tenham sido refrescadas. No modo de refresh distribuído, o refrescamento da linha é intercalado com as operações normais na memória.

O método mais universal para refrescar uma DRAM é o **refresh somente com $\overline{RAS}$**. Ele é realizado amostrando um endereço de linha com $\overline{RAS}$ enquanto $\overline{CAS}$ e $R/\overline{W}$ permanecem em ALTO. A Fig. 11-38 ilustra como esse método é utilizado para um refresh em rajada da TMS44100. Parte da complexidade do arranjo de memória desse chip existe para tornar as operações de refresh mais simples. Como dois bancos estão enfileirados na mesma linha, ambos podem ser refrescados ao mesmo tempo, fazendo efetivamente o mesmo que seria feito se existissem apenas 1.024 linhas. Um **contador de refresh** é utilizado para fornecer 10 bits de endereço de linha para as entradas de endereço da DRAM, iniciando em 0000000000 (linha 0). $\overline{RAS}$ é pulsado em BAIXO para carregar esse endereço na DRAM, e isso refresca a linha 0 nos dois bancos. O contador é incrementado, e o procedimento é repetido até o endereço 1111111111 (linha 1.023). Para a TMS44100, um refresh em rajada pode ser completado em pouco mais de 113 μs e deve ser repetido pelo menos a cada 16 ms.

Enquanto a idéia de um contador de refresh parece bem simples, devemos perceber que os endereços de linha do contador de refresh não podem interferir com os endereços vindos da CPU durante as operações normais de leitura/escrita. Por isso, os endereços de linha do contador de refresh devem ser multiplexados com os endereços da CPU, de modo que a fonte apropriada de endereços para a DRAM seja ativada nos momentos corretos.

Para desobrigar a CPU do computador de algumas dessas responsabilidades, um chip especial chamado **controlador de memória dinâmica** é utilizado freqüentemente. No mínimo, esse chip realiza a multiplexação de endereços e a geração da seqüência de contagem para o refresh, deixando a geração da temporização dos sinais $\overline{RAS}$, $\overline{CAS}$ e $\overline{MUX}$ para algum outro circuito lógico e à pessoa que configura o computador. Outros controladores DRAM são totalmente automáticos. Suas entradas parecem-se bastante com as entradas de uma RAM estática ou de uma ROM. Eles geram automaticamente a seqüência de refresh para manter a memória, multiplexando os endereços, gerando os sinais de $\overline{RAS}$ e $\overline{CAS}$ e arbitrando o controle da DRAM entre os ciclos de leitura/escrita da CPU e as operações locais de refresh. Nos computadores pessoais atuais, o controlador de memória dinâmica e outros controladores de alto nível são implementados em circuitos integrados de aplicação específica (ASIC — *application-specific integrated circuit*).

A maioria dos chips DRAM produzidos atualmente tem capacidade de refresh no próprio chip, o que elimina a necessidade de fornecer endereços externos de refresh. Um desses métodos, mostrado na Fig. 11-39(a), é chamado de *refresh $\overline{CAS}$ antes de $\overline{RAS}$*. Nesse método, o sinal de $\overline{CAS}$ é acionado em BAIXO primeiro e é mantido BAIXO até

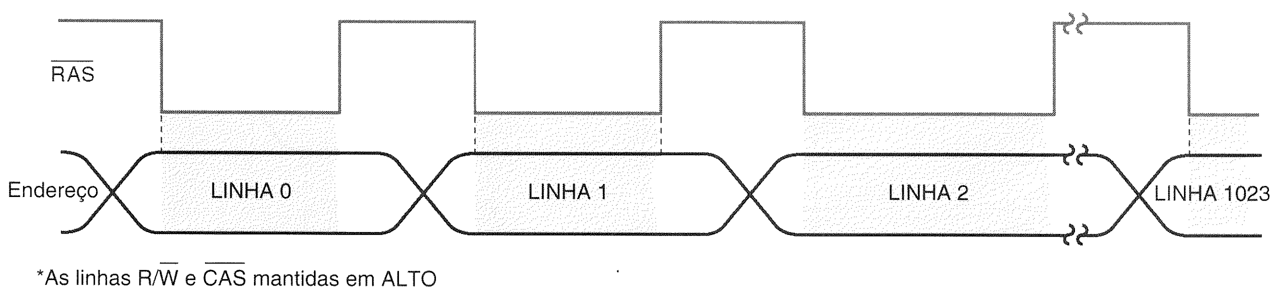


Fig. 11-38 O método de refresh somente com $\overline{RAS}$ utiliza apenas o sinal de $\overline{RAS}$ para carregar o endereço da linha na DRAM para refrescar todas as células nessa linha. O refresh somente com $\overline{RAS}$ pode ser utilizado para realizar um refrescamento em rajada conforme mostrado. Um contador de refresh fornece o endereço seqüencial de linha desde a linha 0 até a linha 1.023 (para uma DRAM de 4M × 1).

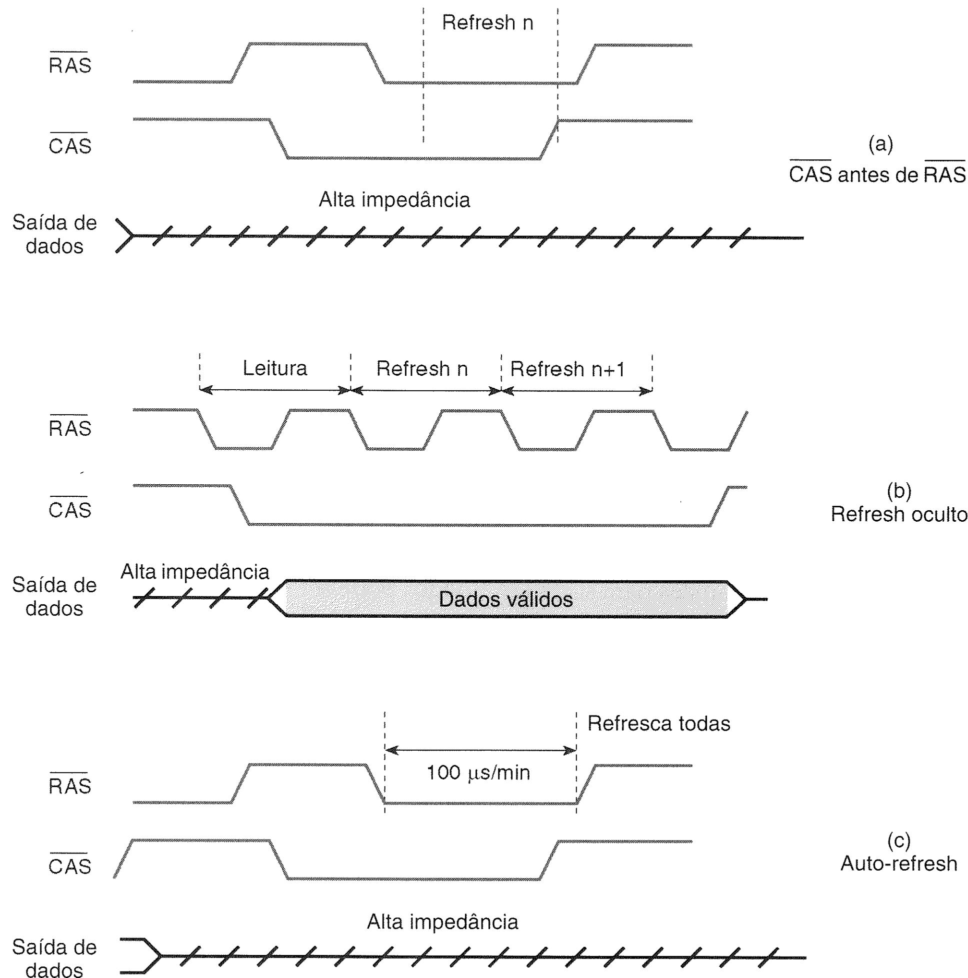


Fig. 11-39 Modos de refresh da TMS44100.

depois de $\overline{RAS}$ ir para BAIXO. Essa seqüência refresca uma linha do arranjo de memória e incrementa um contador interno que gera os endereços de linha. Para realizar um refresh em rajada utilizando essa característica, $\overline{CAS}$ pode ser mantido em BAIXO enquanto $\overline{RAS}$ é pulsado uma vez para cada linha até que todas tenham sido refrescadas. Durante esse ciclo de refresh, todos os endereços externos são ignorados. A TMS44100 também oferece "refresh oculto", que permite que uma linha seja refrescada enquanto os dados são mantidos na saída. Isso é feito mantendo-se $\overline{CAS}$ em BAIXO depois de um ciclo de leitura e então pulsando-se $\overline{RAS}$ conforme a Fig. 11-39(b).

O modo de auto-refresh da Fig. 11-39(c) automatiza totalmente o procedimento. Forçando $\overline{CAS}$ em nível BAIXO antes de $\overline{RAS}$ e então mantendo ambos em BAIXO por pelo menos 100 μs, um oscilador interno aciona o contador de endereços de linha até que todas as células tenham sido refrescadas. O modo que o projetista escolhe depende da taxa de ocupação da CPU do computador. Se ela puder dispor de 100 μs sem acessar sua memória, e se puder fazer isso a cada 16 ms, então o auto-refresh é a melhor escolha. Entretanto, se isso tornar a execução do programa muito mais lenta, pode ser necessário utilizar refresh distribuído com $\overline{CAS}$ antes de $\overline{RAS}$ ou ciclos de refresh oculto. Em qualquer caso, todas as células devem ser refrescadas dentro do tempo previsto ou os dados serão perdidos.

Questões de Revisão

1. *Verdadeiro ou falso*:
   (a) Na maioria das DRAMs é necessário ler apenas uma única célula em cada linha de modo a refrescar todas as células nessa linha.
   (b) No modo de refresh em rajada, o arranjo inteiro é refrescado por um pulso de $\overline{RAS}$.
2. Qual é a função de um contador de refresh?
3. Que funções realiza um controlador de memória dinâmica?
4. *Verdadeiro ou falso*:
   (a) No método de refresh somente com $\overline{RAS}$, o sinal de $\overline{CAS}$ é mantido em BAIXO.
   (b) O refresh do tipo $\overline{CAS}$ antes de $\overline{RAS}$ só pode ser utilizado por DRAMs com circuito de controle de refresh no chip.

11-18 TECNOLOGIA DA RAM DINÂMICA*

Na seleção de um tipo particular de dispositivo RAM para um sistema, o projetista se depara com algumas decisões difíceis. A capacidade (a maior possível), a velocidade (a maior possível), a necessidade de potência (a menor possível), o custo (o menor possível) e a versatilidade (tão fácil de alterar quanto possível) devem ser todos avaliados, pois nenhum tipo de RAM pode maximizar todas essas características desejadas. O mercado de RAM semicondutora está constantemente tentando produzir um misto ideal dessas características em seus produtos para diversas aplicações. Esta seção explica alguns dos termos atuais usados em relação à tecnologia RAM. Este é um tópico muito dinâmico e talvez alguns desses termos estejam ultrapassados antes que o livro tenha sido publicado, mas representam o estado da arte hoje.

SIMMs e DIMMs

Com muitas empresas fabricando placas-mães para sistemas de computadores pessoais, foram adotados conectores padronizados de interface à memória. Esses conectores recebem um pequeno cartão de circuito impresso com pontos de contato em ambos os lados da borda do cartão. Esses cartões modulares permitem uma fácil instalação ou a substituição dos componentes de memória no computador. O módulo de memória em linha simples (SIMM — *single-in-line memory module*) é um cartão de circuito impresso com 72 contatos funcionalmente equivalentes em ambos os lados do cartão. Estes módulos utilizam apenas chips DRAM de 5 V, que variam de capacidade desde 1 até 16 Mbits em encapsulamentos para montagem em superfície do tipo *gull-wing* ou com pinos *J*. Os módulos de memória variam sua capacidade desde 1 até 32 Mbytes.

Os novos módulos de memória em duas linhas (DIMM — *dual-in-line memory module*), de 168 pinos, possuem 84 contatos funcionalmente únicos de cada lado do cartão. Versões de 3,3 V e 5 V estão disponíveis. A capacidade do módulo depende dos chips DRAM que são montados nele; e, conforme a capacidade das DRAMs aumenta, a capacidade dos DIMMs aumenta. Para aplicações compactas, tais como computadores laptop, módulos de memória em duas linhas de baixo perfil estão disponíveis (SODIMM — *small-outline dual-in-line memory module*).

O problema fundamental na indústria de computadores pessoais é fornecer um sistema de memória que seja veloz o suficiente para suportar a crescente taxa de clock dos microprocessadores enquanto mantém os custos num nível razoável. Características especiais estão sendo adicionadas aos dispositivos DRAM básicos para incrementar sua largura de banda total. Embora esses métodos de aumentar a performance estejam constantemente mudando, os termos a seguir estão sendo extensamente usados hoje na literatura relacionada com memórias.

DRAM FPM

O FPM (*fast page mode* — modo de página rápido) permite o acesso mais rápido a qualquer posição de memória dentro da "página" corrente. Uma página é simplesmente uma faixa de endereços de memória que possui os bits mais altos de endereço iguais. Para acessar os dados na página corrente, apenas as linhas mais baixas do endereço precisam ser alteradas. As DRAMs do tipo FPM têm sido fundamentais na indústria de DRAMs nos últimos anos.

DRAM EDO

As DRAMs do tipo EDO (*extended data output* — saída de dados estendida) apresentam uma pequena melhoria em relação às do tipo FPM. Para os acessos numa determinada página, o dado da posição de memória corrente é detectado e armazenado nos pinos de saída. Nas DRAMs FPM, o amplificador sensor aciona a saída sem um latch, requisitando que $\overline{CAS}$ permaneça baixo até que os dados se tornem válidos. Na EDO, enquanto esse dado está presente na saída, $\overline{CAS}$ pode completar seu ciclo, um novo endereço na página corrente pode ser decodificado, e o caminho dos dados pode ser inicializado para o próximo acesso. Isso permite ao controlador da memória enviar o próximo endereço ao mesmo tempo que a palavra corrente está sendo lida.

DRAM BEDO

As DRAMs do tipo BEDO (*burst EDO* — EDO de rajada) levam vantagem pelo fato de a maioria dos dados na memória estar sendo acessada seqüencialmente. A origem menos aleatória desse modo de acesso permite que os dados sejam entregues numa rajada de um, dois, quatro ou oito posições de memória seqüenciais, uma logo após a outra. A primeira posição a ser acessada é a mais lenta devido ao procedimento de armazenar os endereços de linha e de coluna. Entretanto, as diversas posições a seguir podem ser acessadas muito rapidamente gerando-se os endereços mais baixos (da coluna) dentro do chip de memória, em vez de armazená-los do barramento externo. Esses dispositivos necessitam de um clock para incrementar o contador de endereços e amostrar o dado do chip. A ação do clock é realizada pulsando-se a linha de $\overline{CAS}$ para ler sucessivamente os dados da RAM. Enquanto o último dado da rajada estiver sendo enviado pelo $\overline{CAS}$, um novo endereço de coluna pode ser amostrado sem a inclusão de estados de espera adicionais. Isso permite um fluxo contínuo de dados para serem lidos da página corrente na maior taxa possível.

SDRAM

As DRAMs síncronas (*synchronous* DRAM) parecem ser a grande promessa como o próximo padrão de sistemas de alto desempenho. Os dados são gerados pelo clock do sistema (em vez da linha de controle $\overline{CAS}$) em rajadas de posições de memória dentro da mesma página. Internamente, as SDRAMs são organizadas em dois bancos. Isso permite que os dados sejam lidos a taxas bastante elevadas, acessando-se alternadamente cada um dos dois bancos. Para fornecer todas as características e a flexibilidade necessária para esse tipo de DRAM trabalhar com uma ampla variedade de sistemas, os circuitos internos da SDRAM têm se tornado mais complexos. Uma seqüência de comandos é necessária para informar à SDRAM que opções são necessári-

* Este tópico pode ser omitido sem afetar a continuidade do restante do livro.

as, tais como tamanho da rajada, dados seqüenciais ou intercalados e modo de refresh $\overline{CAS}$ antes de $\overline{RAS}$ ou auto-refresh. O modo de auto-refresh permite ao dispositivo de memória realizar todas as funções necessárias para manter suas células refrescadas. As SDRAMs naturalmente são mais caras, mas reduzem a necessidade de circuitos de suporte. Isso torna o custo total do sistema que utiliza SDRAMs semelhante ao dos outros sistemas anteriores.

Questões de Revisão

1. SIMMs e DIMMs são intercambiáveis?
2. O que é uma "página" de memória?
3. Por que o "modo página" é mais rápido?
4. O que significa o termo *EDO*?
5. Qual é o termo utilizado para acessar várias posições consecutivas de memória?
6. Com que uma SDRAM está sincronizada?

11-19 EXPANSÃO DO TAMANHO DA PALAVRA E DA CAPACIDADE

Em muitas aplicações de memória, o tamanho da palavra ou a capacidade necessária de RAM ou ROM não podem ser satisfeitos por apenas um chip de memória. Em vez disso, vários chips de memória devem ser combinados para fornecer a capacidade e/ou tamanho da palavra. Veremos como isso é feito através de diversos exemplos, que ilustram as idéias principais que são usadas quando os chips de memória são interfaceados com um microprocessador. Os exemplos a seguir pretendem ser instrutivos, e os tamanhos dos chips de memória utilizados foram escolhidos para conservar espaço. As mesmas técnicas apresentadas podem ser estendidas para chips de memória de maior capacidade.

Expansão do Tamanho da Palavra

Suponha que precisamos de uma memória que possa armazenar 16 palavras de oito bits e que tudo que temos são chips de RAM de 16 × 4 com linhas de entrada e saída comuns. Podemos combinar dois desses chips de 16 × 4 para implementar a memória desejada. A configuração para fazer isso é mostrada na Fig. 11-40. Examine esse diagrama cuidadosamente e veja o que você pode descobrir sobre ele antes de prosseguir a leitura.

Como cada chip pode armazenar 16 palavras de quatro bits e queremos armazenar 16 palavras de oito bits, utilizamos cada chip para armazenar a *metade* de cada palavra. Em outras palavras, RAM-0 armazena os quatro bits de *mais alta* ordem de cada uma das 16 palavras, e RAM-1 armaze-

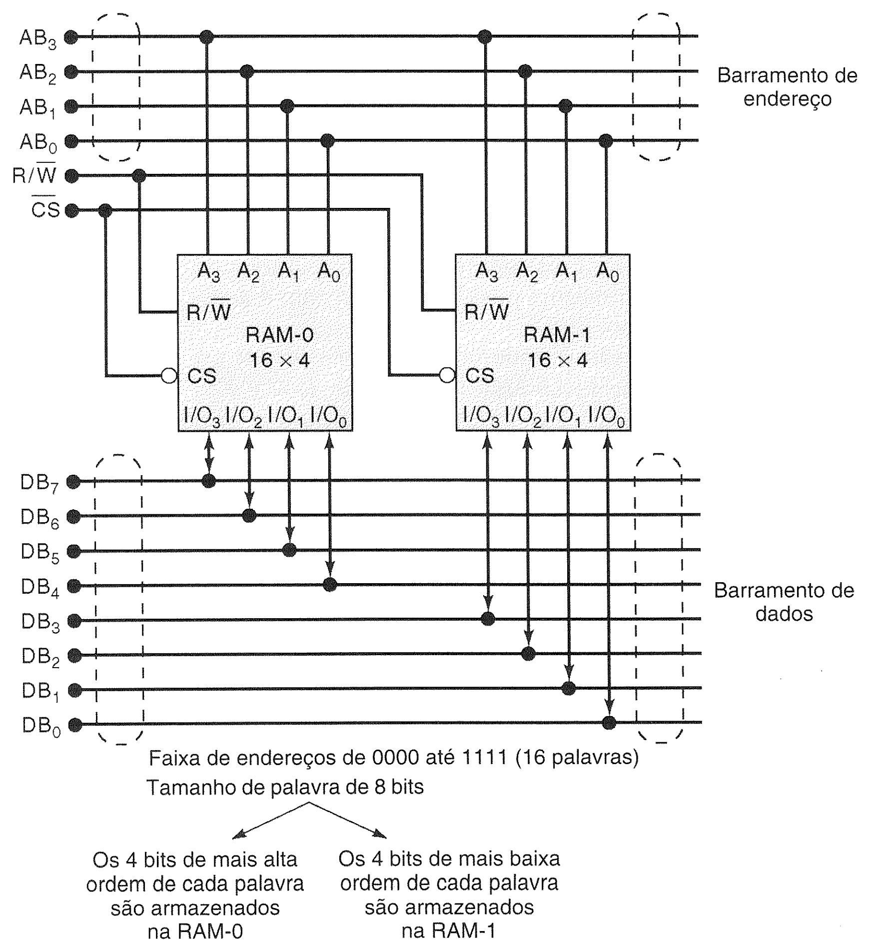


Fig. 11-40 Combinando duas RAMs de 16 × 4 para um módulo de 16 × 8.

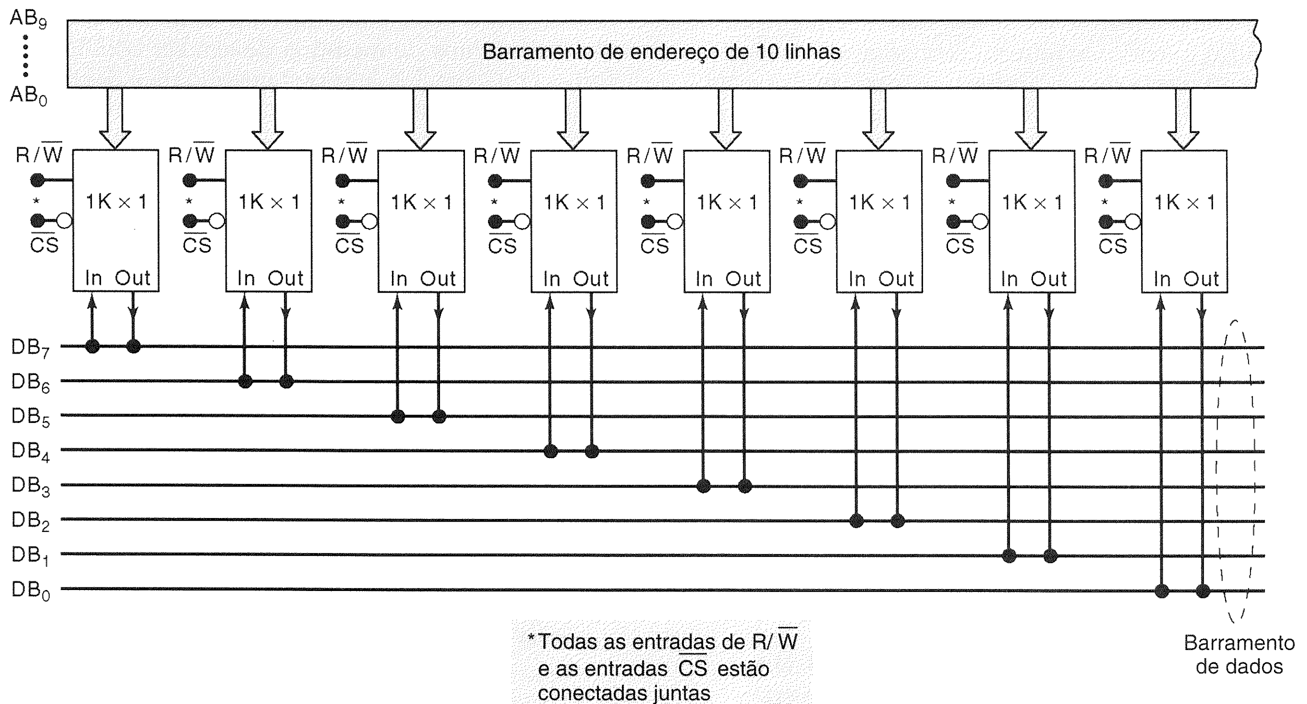


Fig. 11-41 Oito chips 2125A de 1K × 1 interligados como uma memória de 1K × 8.

na os quatro bits de *mais baixa* ordem de cada uma das 16 palavras. Uma palavra de oito bits completa está disponível nas saídas da RAM conectadas no barramento de dados.

Qualquer uma das 16 palavras é selecionada aplicando-se o código de endereço apropriado no *barramento de endereço* de quatro linhas (AB_3, AB_2, AB_1 e AB_0). As linhas de endereço tipicamente vêm da CPU. Note que cada linha do barramento de endereço é conectada na entrada de endereço correspondente de cada chip. Isso significa que, uma vez que o código de endereço é colocado nesse barramento, esse mesmo código de endereço é aplicado a ambos os chips, de modo que a mesma posição de cada chip é acessada ao mesmo tempo.

Uma vez selecionado o endereço, podemos ler ou escrever dele sob o controle das linhas comuns de $R/\overline{W}$ e $\overline{CS}$. Para ler, $R/\overline{W}$ deve estar em ALTO e $\overline{CS}$ deve estar BAIXO. Isso faz as linhas de entrada e saída da RAM atuarem como *saídas*. A RAM-0 coloca sua palavra de quatro bits selecionada nas quatro linhas mais altas do barramento de dados, e a RAM-1 coloca sua palavra de quatro bits selecionada nas quatro linhas mais baixas do barramento de dados. Desse modo, o barramento de dados contém a palavra completa de oito bits selecionada, que agora pode ser transmitida para algum outro dispositivo (usualmente um registrador na CPU).

Para escrita, $R/\overline{W} = 0$ e $\overline{CS} = 0$ fazem com que as linhas de entrada e saída operem como *entradas*. A palavra de oito bits a ser escrita é colocada no barramento de dados (usualmente pela CPU). Os quatro bits mais altos serão escritos na posição selecionada da RAM-0, e os quatro bits mais baixos serão escritos na RAM-1.

Em resumo, a combinação dos dois chips de RAM atua como um único chip de memória de 16 × 8. Podemos nos referir a essa combinação como um *módulo de memória* de 16 × 8.

A mesma idéia básica para expandir o tamanho da palavra funciona em muitas situações diferentes. Leia o exemplo a seguir e esboce um diagrama de como o sistema será, antes de olhar a solução apresentada.

EXEMPLO 11-13

A 2125A é um CI de RAM estática que tem capacidade de 1K × 1, uma entrada de seleção ativa em BAIXO e entradas e saídas de dados separadas. Mostre como combinar diversos CIs 2125A para formar um módulo de 1K × 8.

Solução

A configuração é mostrada na Fig. 11-41, onde oito chips 2125A são utilizados para formar um módulo de 1K × 8. Cada chip armazena um dos bits de cada uma das 1.024 palavras de oito bits. Repare que todas as entradas de $R/\overline{W}$ e de $\overline{CS}$ estão ligadas juntas, e o barramento de endereço de 10 linhas está conectado nas entradas de endereço de cada chip. Note também que como o 2125A tem pinos de entrada e saída de dados separados, e os pinos de cada chip são ligados na mesma linha do barramento de dados.

Expansão da Capacidade

Suponha que precisamos de uma memória que possa armazenar 32 palavras de quatro bits e temos chips de 16 × 4. Combinando-se dois chips de 16 × 4 conforme mostra-

do na Fig. 11-42, podemos produzir a memória desejada. Mais uma vez, examine esse diagrama e veja o que você pode descobrir antes de prosseguir com a leitura.

Cada RAM é utilizada para armazenar 16 palavras de quatro bits. Os quatro pinos de entrada e saída de dados de cada RAM estão conectados ao barramento de dados comum de quatro linhas. Somente um chip de RAM pode ser selecionado (habilitado) de cada vez, de modo a que não haja problemas de contenção de barramento. Isto é garantido acionando-se as respectivas entradas $\overline{CS}$ com sinais lógicos diferentes.

Como a capacidade total desse módulo é de 32 × 4, devem existir 32 endereços diferentes. Isso requer *cinco* linhas do barramento de endereço. A linha de endereço mais alta, AB_4, é utilizada para selecionar uma ou outra RAM (através das entradas $\overline{CS}$) como aquela que será lida ou escrita. As outras quatro linhas de endereço, AB_0 a AB_3, são usadas para selecionar uma posição de memória dentre as 16 do chip de RAM selecionado.

Para ilustrar, quando $AB_4 = 0$, o $\overline{CS}$ da RAM-0 habilita esse chip para leitura ou escrita. Então, qualquer posição da RAM-0 pode ser acessada por AB_3 a AB_0. As quatro últimas linhas de endereço podem variar de 0000 até 1111 para selecionar a posição desejada. Desse modo, a faixa de endereços representando posições da RAM-0 é

$$AB_4AB_3AB_2AB_1AB_0 = 00000 \text{ até } 01111$$

Repare que, quando $AB_4 = 0$, o $\overline{CS}$ da RAM-1 está ALTO, de modo que suas linhas de entrada e saída estão desabilitadas (alta impedância) e ela não pode se comunicar (fornecer dados ou pegar dados) com o barramento de dados.

Deve ficar claro que, quando $AB_4 = 1$, os papéis de RAM-0 e RAM-1 são trocados. RAM-1 agora fica habilitada e as linhas de AB_3 a AB_0 selecionam uma de suas posições. Portanto, a faixa de endereços localizados na RAM-1 é

$$AB_4AB_3AB_2AB_1AB_0 = 10000 \text{ até } 11111$$

EXEMPLO 11-14

Deseja-se associar diversas PROMs de 2K × 8 para produzir uma capacidade total de 8K × 8. Quantos chips de PROM são necessários? Quantas linhas de endereço são necessárias?

Solução

Quatro chips PROM são necessários, com cada um armazenando 2K palavras das 8K. Já que 8K = 8 × 1.024 = 8.192 = 2^{13}, *treze* linhas de endereço são necessárias.

A configuração para a memória do Exemplo 11-14 é semelhante à da memória de 32 × 4 da Fig. 11-42. Entretanto, ela é ligeiramente mais complexa, pois requer um circuito decodificador para a geração dos sinais de entrada $\overline{CS}$. O diagrama completo para essa memória de 8.192 × 8 está mostrado na Fig. 11-43.

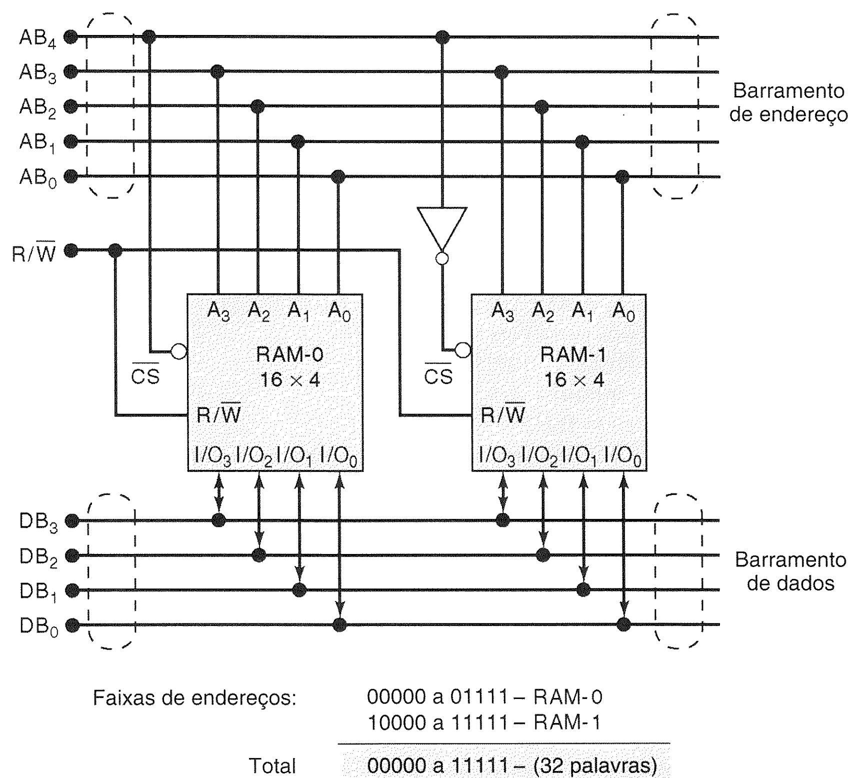


Fig. 11-42 Combinando dois chips de 16 × 4 para formar uma memória de 32 × 4.

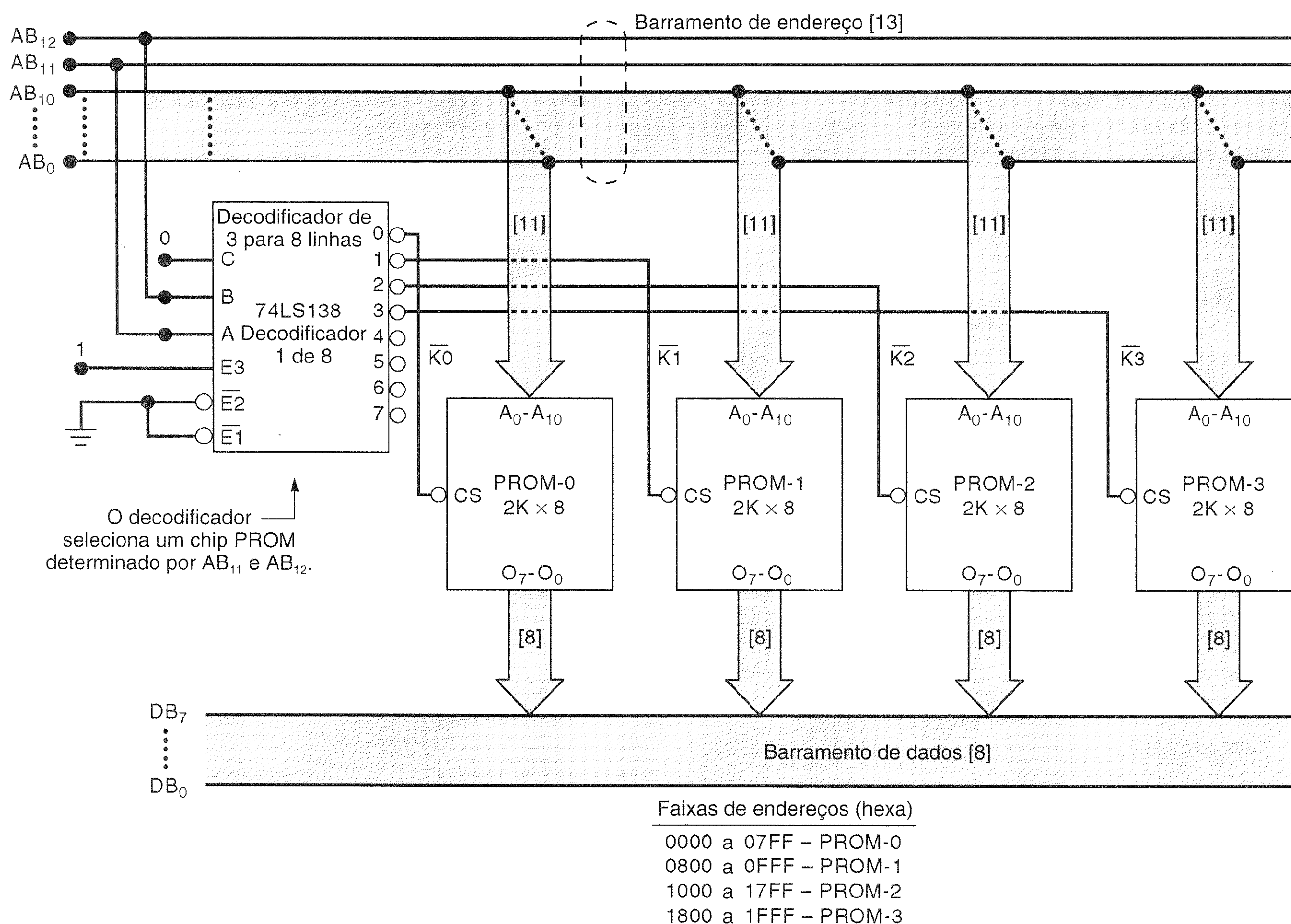


Fig. 11-43 Quatro PROMs de 2K × 8 utilizadas para obter uma capacidade total de 8K × 8.

Tendo em vista que a capacidade total é de 8.192 palavras, 13 linhas do barramento de endereço são necessárias. As duas linhas de mais alta ordem, AB_{11} e AB_{12}, são usadas para selecionar *um* dos chips de PROM; as outras 11 linhas do barramento de endereço vão para cada PROM para selecionar a posição desejada dentro da PROM selecionada. A seleção da PROM é realizada levando AB_{11} e AB_{12} para o decodificador 74LS138. As quatro combinações possíveis são decodificadas para gerar sinais ativos em BAIXO, que são aplicados nas entradas $\overline{CS}$. Por exemplo, quando $AB_{11} = AB_{12} = 0$, a saída K0 do decodificador vai para BAIXO (todas as outras ficam em ALTO) e habilita a PROM-0. Isso faz com que as saídas da PROM-0 forneçam a palavra de dados internamente armazenada no endereço determinado por AB_0 a AB_{10}. Todas as outras PROMs estão desabilitadas, de modo que não há contenção de barramento.

Enquanto $AB_{12} = AB_{11} = 0$, os valores de AB_{10} a AB_0 podem variar desde tudo 0 até tudo 1. Assim, a PROM-0 responde à seguinte faixa de endereços:

$$AB_{12} - AB_0 = 0000000000000 \text{ até } 0011111111111$$

Por conveniência, esses endereços podem ser expressos mais facilmente no código hexadecimal para representar uma faixa de 0000 a 07FF.

Analogamente, quando $AB_{12} = 0$ e $AB_{11} = 1$, o decodificador seleciona a PROM-1, que por sua vez responde fornecendo a palavra que tem armazenada internamente no endereço codificado por AB_{10} até AB_0. Logo, PROM-1 responde à seguinte faixa de endereços:

0100000000000 até 0111111111111 (binário)

ou

0800 a 0FFF (hexa)

Você deve verificar as faixas de endereços de PROM-2 e PROM-3 apresentadas na figura.

Obviamente, as linhas de endereço AB_{12} e AB_{11} são utilizadas para selecionar um dos quatro chips de PROM, enquanto AB_{10} a AB_0 selecionam a palavra armazenada na PROM selecionada.

EXEMPLO 11-15

Que tipo de decodificador seria necessário para expandir a memória da Fig. 11-43 para 32K × 8? Descreva que linhas de endereço são usadas.

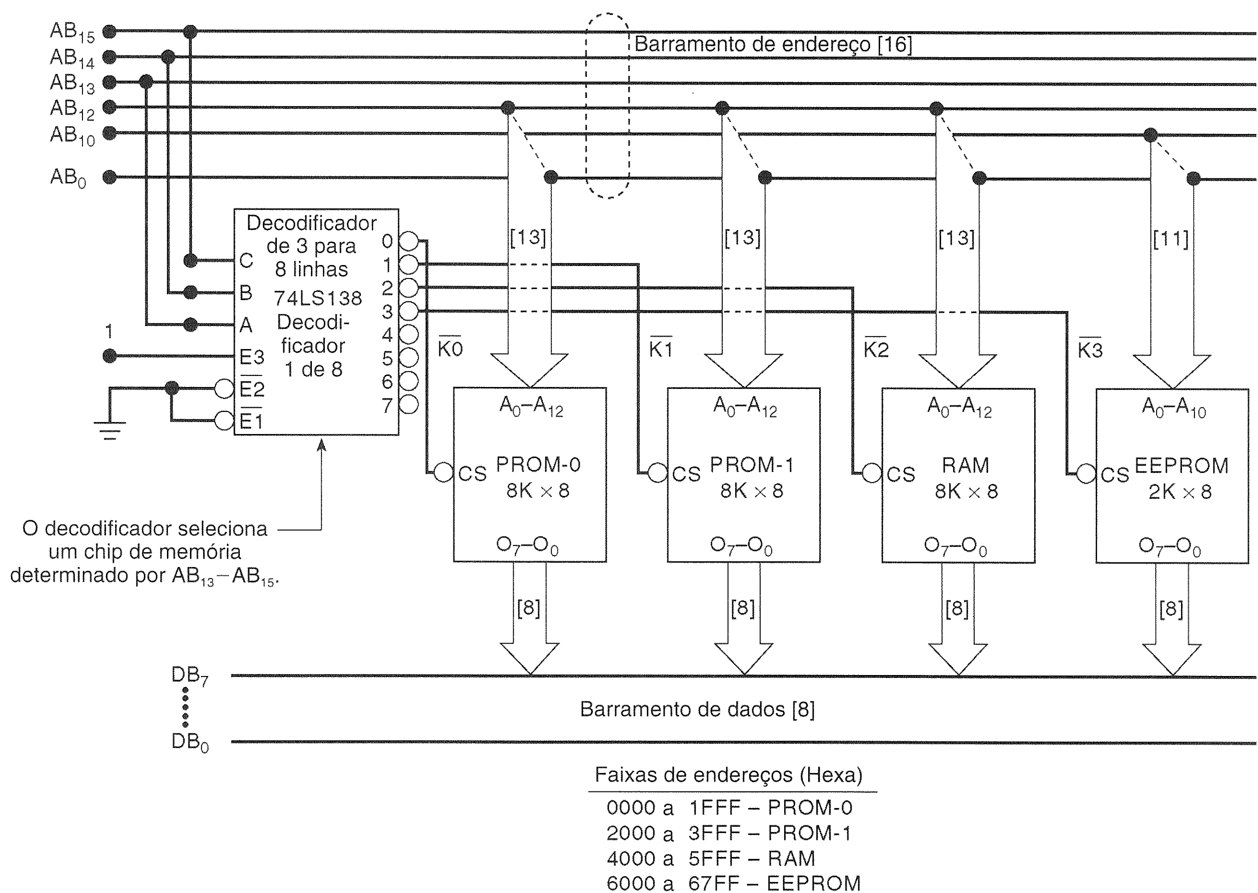


Fig. 11-44 Um sistema com decodificação de endereços incompleta.

Solução

Para uma capacidade de 32K, seriam necessários 16 chips de PROM. Para selecionar uma das 16 PROMs, seria necessário um decodificador de 4 para 16 linhas. Quatro linhas de endereço (AB_{14}, AB_{13}, AB_{12}, AB_{11}) seriam conectadas como entradas do decodificador. As linhas de endereço AB_{10} a AB_0 seriam conectadas nas entradas de endereço de cada uma das 16 PROMs. Assim, um total de 15 linhas de endereço seria usado. Isso concorda com o fato de que 2^{15} = 32.768 = 32K.

Decodificação Incompleta de Endereços

Em muitas situações, existe a necessidade de utilizar vários dispositivos de memória num mesmo sistema de memória. Por exemplo, considere os requisitos de um sistema de painel digital num automóvel. Tal sistema geralmente é implementado utilizando-se um microprocessador. Conseqüentemente, precisamos de alguma ROM não-volátil para armazenar as instruções do programa. Precisamos de alguma memória de leitura/escrita para armazenar os dígitos que representam a velocidade, as RPMs, a quantidade de combustível e assim por diante. Outros valores digitalizados devem ser armazenados para representar a pressão do óleo, a temperatura da máquina, a tensão da bateria e assim por diante. Também precisamos de algum armazenamento não-volátil de leitura/escrita (EEPROM) para o valor do odômetro, já que não seria bom que esse número fosse ressetado para 0 ou assumisse um valor qualquer sempre que a bateria do carro fosse desconectada.

A Fig. 11-44 mostra um sistema de memória que poderia ser utilizado em um sistema de microcomputador. Note que a parte ROM é construída com dois dispositivos de 8K × 8 (PROM-0 e PROM-1). A seção de RAM requisita um único dispositivo de 8K × 8. A EEPROM disponível é um dispositivo de apenas 2K × 8. O sistema de memória necessita de um decodificador para selecionar somente um dispositivo de cada vez. Esse decodificador divide o espaço de endereçamento total (supondo endereços de 16 bits) em blocos de 8K endereços. Em outras palavras, cada saída do decodificador é ativada por 8.192 (8K) endereços diferentes. Repare que as três linhas de endereço mais altas controlam o decodificador. As 13 linhas de endereço de mais baixa ordem estão ligadas diretamente nas entradas de endereço dos chips de memória. A única exceção a isso é a EEPROM que tem somente 11 linhas de endereço para sua capacidade de 2 Kbytes. Se o endereço (em hexa) dessa EEPROM deve ficar na faixa de 6000 a 67FF, ela responderá a esses endereços conforme planejado. Entretanto, as duas linhas de endereços, A_{11} e A_{12}, não estão presentes no esquema de decodificação para esse chip. A saída $K3$ do decodificador está ativa para 8K endereços, mas o chip que está conectado contém apenas 2K posições. Em resultado, a EEPROM tam-

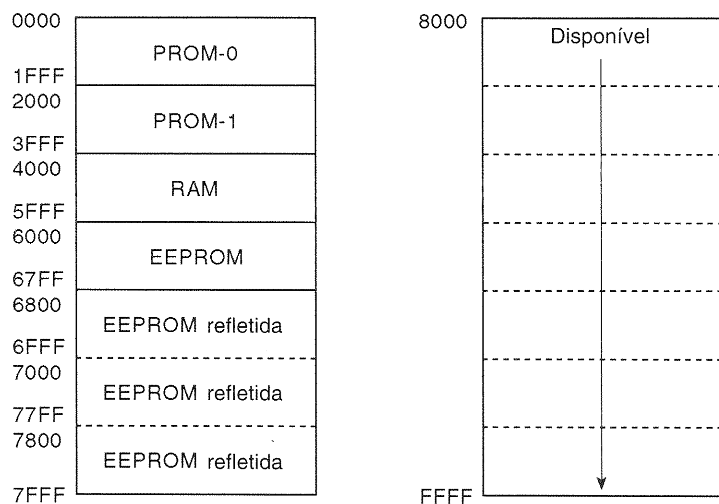


Fig. 11-45 Um mapa de memória de um painel digital.

bém responderá aos outros 6K endereços desse bloco decodificado de memória. O mesmo conteúdo da EEPROM vai aparecer nos endereços 6800-6FFF, 7000-77FF e 7800-7FFF. Essas áreas de memória que são ocupadas de modo redundante pelo dispositivo devido à decodificação incompleta de endereços são referenciadas como áreas de **memória refletida**. Isso ocorre freqüentemente em sistemas onde existem uma abundância de espaço de endereçamento e uma necessidade de minimizar a lógica de decodificação. Um **mapa de memória** desse sistema, conforme apresentado na Fig. 11-45, mostra claramente os endereços atribuídos a cada dispositivo, assim como o espaço de memória que está disponível para expansão.

Combinando Chips de DRAM

CIs de DRAM freqüentemente são fornecidos com tamanho de palavra de um a quatro bits, de modo que é necessário combinar vários deles para formar módulos com tamanhos de palavra maiores. A Fig. 11-46 mostra como combinar oito chips DRAM TMS44100 para formar um módulo de 4M × 8. Cada chip tem capacidade de 4M × 1.

Existem vários pontos importantes a notar. Primeiro, como $4M = 2^{22}$, o chip TMS44100 possui *onze* entradas de endereço; lembre-se de que as DRAMs utilizam entradas multiplexadas de endereços. O multiplexador de endereços recebe as 22 linhas de endereço da CPU e as transforma em um barramento de endereço de 11 linhas para os chips DRAM. Segundo, as entradas de $\overline{RAS}$, $\overline{CAS}$ e $\overline{WE}$ de todos os oito chips são conectadas em comum, de modo que todos os chips são ativados simultaneamente para cada operação de memória. Finalmente, lembre-se de que o TMS44100 possui circuito de refresh interno, e portanto não há necessidade de um controlador de DRAM externo.

Questões de Revisão

1. O MCM6209C é um chip de RAM estática de 64K × 4. Quantos destes chips são necessários para formar um módulo de 1M × 4?
2. Quantos são necessários para um módulo de 64K × 16?
3. *Verdadeiro ou falso*: Quando chips de memória são combinados para formar um módulo com tamanho de palavra ou capacidade maior, as entradas *CS* de cada chip sempre são conectadas juntas.
4. *Verdadeiro ou falso*: Quando chips de memória são combinados para formar um módulo com capacidade maior, cada chip é conectado nas mesmas linhas do barramento de dados.

11-20 FUNÇÕES ESPECIAIS DA MEMÓRIA

Vimos que dispositivos RAM e ROM são utilizados como memória interna de alto desempenho do computador que se comunicam diretamente com a CPU (por exemplo, microprocessador). Nesta seção, descrevemos sucintamente algumas funções especiais que dispositivos de memória semicondutora realizam em computadores e em outros sistemas e equipamentos digitais. A apresentação não pretende fornecer detalhes de como essas funções são implementadas, mas introduzir as idéias básicas.

Armazenamento de Dados com o Sistema Desligado

Em muitas aplicações, a volatilidade da RAM semicondutora pode significar a perda de dados críticos quando a energia do sistema é desligada, propositadamente ou como resultado de falta de energia. Para citar apenas dois exemplos:

1. Parâmetros críticos de operação de terminais gráficos, terminais inteligentes e impressoras. Esses parâmetros alteráveis determinam os modos de operação e atributos que serão válidos quando o sistema está ligado.
2. Sistemas de controle de processos industriais que nunca "se perdem" no meio de uma tarefa quando a energia falta inesperadamente.

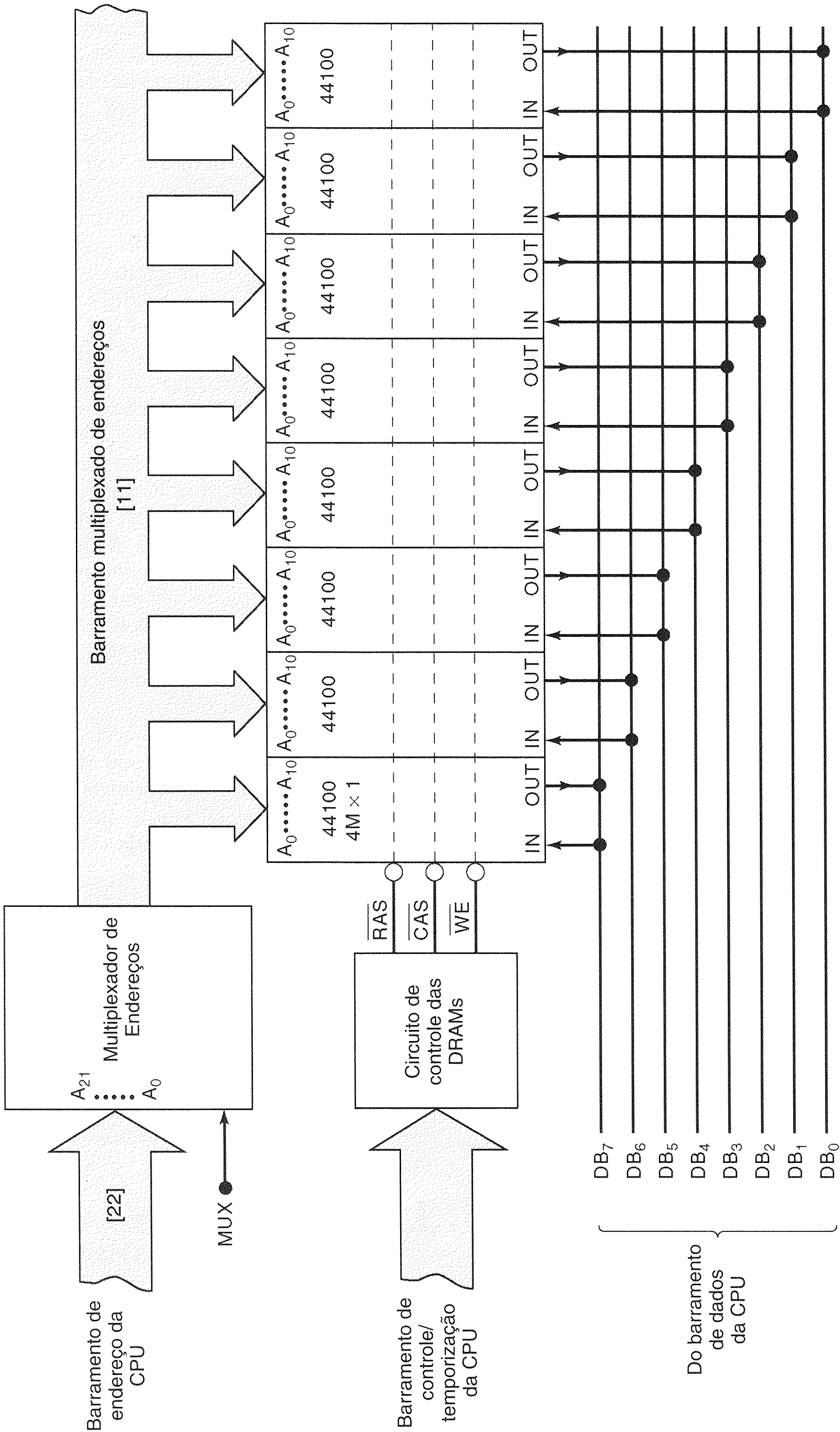


Fig. 11-46 Oito chips DRAM de 4M × 1 combinados para formar um módulo de memória de 4M × 8.

Existem muitos modos de realizar o armazenamento de dados críticos nas situações de falta de energia. Num método, todos os dados críticos durante a operação normal do sistema são armazenados em RAM, que pode ser alimentada por baterias de reserva sempre que a energia for interrompida. Alguns chips de RAM CMOS possuem requisitos de energia muito reduzidos em modo de espera (tão baixos quanto 0,5 mW) e são particularmente apropriados para essa tarefa. Algumas SRAMs CMOS incluem uma pequena bateria de lítio diretamente com o chip. Naturalmente, mesmo com seu baixo consumo de energia, essas RAMs CMOS eventualmente consomem as baterias se a energia ficar desligada por períodos prolongados, e os dados serão perdidos.

Outra técnica armazena todos os dados críticos do sistema em memória flash não-volátil. Esse método tem a vantagem de não necessitar de bateria, e portanto não representa risco de perda de dados mesmo por períodos prolongados de falta de energia. As memórias flash, entretanto, não podem ter seus dados alterados tão facilmente como numa RAM estática. Lembre-se de que com um chip flash não podemos apagar e escrever em um ou dois bytes; essa operação deve ser feita num setor de cada vez (usualmente 512 bytes). Isso requer que a CPU tenha que reescrever um grande bloco de dados mesmo quando somente uns poucos bytes necessitam ser alterados.

Numa terceira técnica, a CPU armazena todos os dados em memória RAM volátil de alto desempenho durante a operação normal do sistema. Na falta de energia, a CPU executa um pequeno programa de falta de energia (a partir da ROM) que transfere os dados críticos da RAM do sistema para uma RAM CMOS socorrida a bateria ou para uma memória flash não-volátil. Isso requer um circuito especial, que detecta a iminência de uma falta de energia e envia um sinal para a CPU para informá-la de que deve executar a seqüência apropriada para falta de energia.

Em qualquer caso, quando a energia é restaurada, a CPU executa um programa de inicialização (a partir da ROM) que transfere os dados críticos da memória de armazenamento de backup para a RAM do sistema, de modo que o sistema possa prosseguir sua operação de onde estava quando a energia foi interrompida.

Memória Cache

Computadores e outros sistemas digitais podem ter milhares ou milhões de bytes de memória interna (RAM ou ROM) que armazena programas e dados de que a CPU necessita durante sua operação normal. Normalmente, isso obrigaria toda a memória interna a ter uma velocidade de operação comparável à da CPU, de modo a alcançar o maior desempenho possível do sistema. Em muitos sistemas não é econômico utilizar dispositivos de memória de alta velocidade para toda a memória interna. Em vez disso, os projetistas usam um pequeno bloco de memória *cache* de alta velocidade que pode ter, digamos, 512 palavras. Este bloco de memória cache é o único bloco que se comunica diretamente com a CPU em alta velocidade; as instruções do programa e os dados são transferidos da memória interna, lenta e barata, para a memória cache quando são necessários para a CPU. O sucesso da memória cache depende de diversos fatores complexos, e alguns sistemas não se beneficiam de usar memória cache.

Memória First-In, First-Out (FIFO)

Em sistemas de memória **FIFO** (*first-in, first-out* — primeiro a entrar, primeiro a sair), os dados que são escritos na área de armazenamento da RAM são lidos na mesma ordem em que foram escritos. Explicando de outro modo, a primeira palavra escrita no bloco de memória é a primeira palavra que é lida do bloco de memória; daí o nome FIFO. Essa idéia está ilustrada na Fig. 11-47.

A parte (a) mostra a seqüência de escrita de três bytes de dados no bloco de memória. Note que, conforme cada novo byte é escrito na posição 1, os outros bytes são movidos para a próxima posição. A parte (b) mostra a seqüência de leitura de dados para fora do bloco FIFO. O primeiro byte lido é o mesmo que o primeiro byte que foi escrito, e assim por diante. A operação da FIFO é controlada por *registradores apontadores* especiais que guardam onde os dados devem ser escritos e de onde eles devem ser lidos.

Uma FIFO é útil como um **buffer de transferência de dados** entre sistemas que transferem dados a taxas bastante diferentes. Um exemplo é a transferência de dados de um computador para uma impressora. O computador envia o caracter para a impressora a uma taxa muito alta, digamos, um byte a cada 10 μs. Esses dados preenchem uma memória FIFO na impressora. A impressora então lê os dados da FIFO numa taxa bem mais baixa, digamos, um byte a cada 5 ms, e imprime os caracteres correspondentes na mesma ordem enviada pelo computador.

Uma FIFO também pode ser usada como um buffer de transferência de dados entre um dispositivo lento, tal como um teclado, e um computador de alta velocidade. Nesse caso, a FIFO aceita os dados do teclado numa taxa baixa e os armazena; o computador pode então ler os dados armazenados do teclado numa taxa mais alta. Desse modo, o computador pode realizar outras tarefas enquanto a FIFO vai sendo lentamente preenchida com os dados.

Questões de Revisão

1. Quais são os diversos modos de tratar a possível perda de dados críticos quando a energia é interrompida?
2. Qual é a principal razão para utilizar uma memória cache?
3. O que significa *FIFO*?
4. O que é um buffer de transferência de dados?

11-21 PESQUISA DE FALHAS EM SISTEMAS COM RAM

Todos os computadores utilizam RAM. Muitos computadores de propósito geral e a maioria dos computadores de propósito específico (tais como os controladores baseados em microprocessadores e os computadores para controle de processos) também utilizam alguma forma de ROM. Cada CI de RAM ou ROM que faz parte da memória interna de um computador contém tipicamente milhares de células de memória. Uma única célula de memória com problemas pode causar uma falha geral no sistema (comumente denominada de "*crash*" do sistema) ou, no mí-

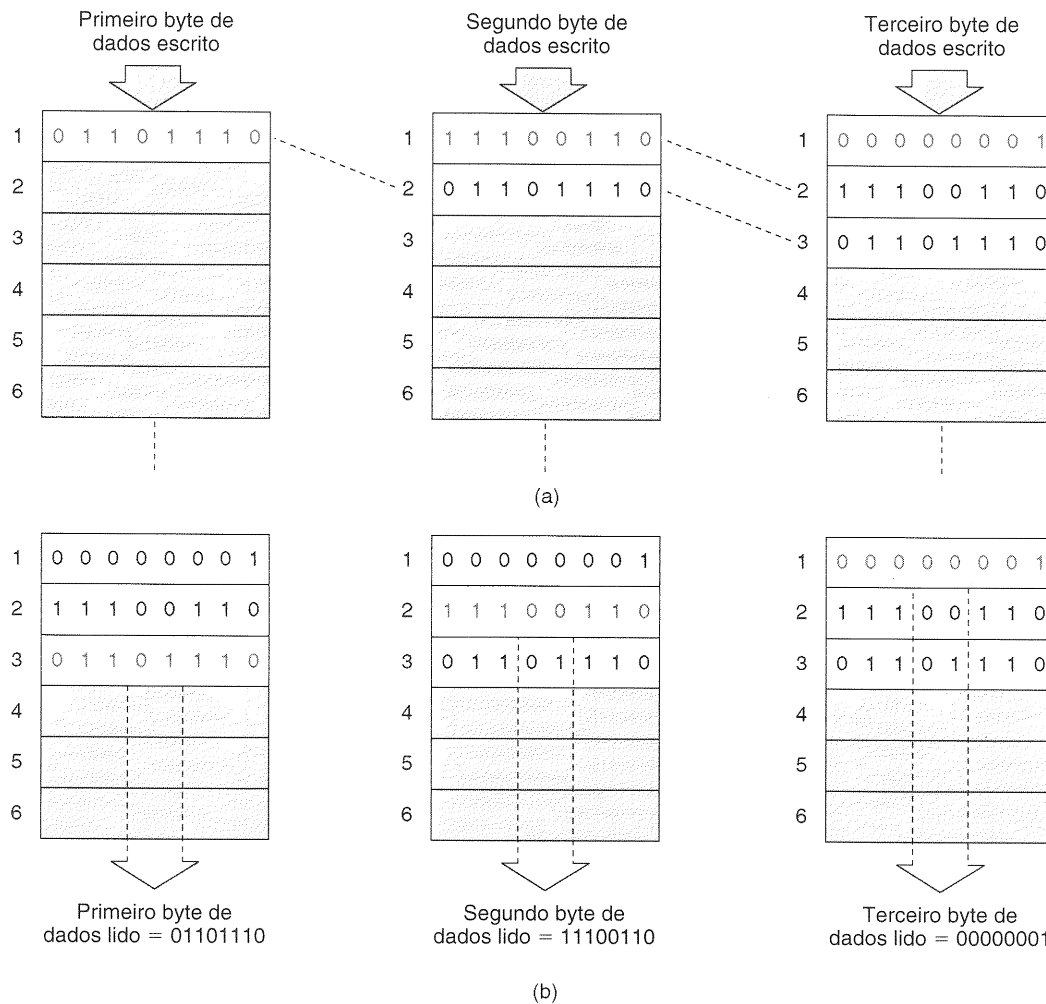


Fig. 11-47 Na FIFO, os valores são lidos da memória (b) na mesma ordem que foram escritos na memória (a).

nimo, uma operação não-confiável do sistema. O teste e a pesquisa de falhas de sistemas de memória envolvem o uso de técnicas que não são usadas freqüentemente em outras partes do sistema digital. Como a memória consiste em milhares de circuitos idênticos atuando como posições de armazenamento, qualquer teste de sua operação deve envolver uma verificação para detectar exatamente quais posições estão operando bem e quais não estão. Logo, analisando o padrão das posições boas e ruins, em conjunto com a organização do circuito de memória, pode-se determinar as possíveis causas de mau funcionamento da memória. Tipicamente, o problema pode ser rastreado até um CI de memória ruim; um CI decodificador, uma porta lógica ou um buffer de sinal ruim; ou um problema nas conexões do circuito (isto é, conexões abertas ou em curto).

Tendo em vista que a RAM deve ser escrita e lida, o teste geralmente é mais complexo do que testar uma ROM. Nesta seção, analisaremos alguns procedimentos comuns para testar a parte RAM da memória e interpretar os resultados dos testes. Examinaremos o teste de ROM na próxima seção.

Conhecer a Operação

O sistema de memória RAM mostrado na Fig. 11-48 será utilizado em nossos exemplos. Conforme enfatizamos em discussões anteriores, o sucesso na pesquisa de falhas de um circuito ou sistema relativamente complexo começa por um conhecimento completo de sua operação. Antes de discutirmos o teste desse sistema com RAM, devemos primeiro analisá-lo cuidadosamente, de modo a compreender sua operação.

A capacidade total é 4K × 8 e é constituída por módulos de RAM de 1K × 8. Um módulo pode ser apenas um único CI, ou pode consistir em vários CIs (por exemplo, dois chips de 1K × 4). Cada módulo está conectado à CPU através dos barramentos de endereço e de dados e da linha de controle $R/\overline{W}$. Os módulos têm linhas de entrada e saída de dados comuns. Durante uma operação de leitura, essas linhas se tornam linhas de saída de dados através das quais o módulo selecionado coloca o dado no barramento para a CPU ler. Durante uma operação de escrita, essas linhas funcionam como linhas de entrada para aceitar o dado gerado no barramento pela CPU e escrevê-lo na posição selecionada.

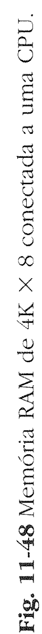


Fig. 11-48 Memória RAM de 4K × 8 conectada a uma CPU.

O decodificador 74LS138 e a porta OR de quatro entradas decodificam as seis linhas de endereços de mais alta ordem para gerar os sinais de seleção dos chips, $\overline{K0}$, $\overline{K1}$, $\overline{K2}$ e $\overline{K3}$. Estes sinais habilitam um módulo específico de RAM para operação de leitura ou escrita. O INVERSOR é utilizado para inverter o sinal de clock de saída da CPU, ϕ_2, de modo que o decodificador é habilitado somente enquanto ϕ_2 está em ALTO. O pulso ϕ_2 ocorre somente depois de permitir tempo suficiente para as linhas de endereço se estabilizarem, quando da aplicação de um novo endereço no barramento. ϕ_2 ficará em BAIXO enquanto as linhas de endereço e de $R/\overline{W}$ estiverem mudando; isso evita a ocorrência de glitches na saída do decodificador que poderiam ativar erradamente um chip de memória e possivelmente causar o armazenamento de dados inválidos.

Cada módulo de RAM possui suas entradas de endereço conectadas nas linhas A_0 a A_9 do barramento de endereço da CPU. As linhas de endereço de mais alta ordem A_{10} até A_{15} selecionam um dos módulos de RAM. O módulo selecionado decodifica as linhas de endereço, A_0 a A_9, para encontrar a posição da palavra que está sendo endereçada. Os exemplos a seguir mostrarão como determinar os endereços que correspondem a cada módulo.

EXEMPLO 11-16

Suponha que a CPU esteja realizando uma operação de leitura no endereço 06A3 (hexa). Qual módulo de RAM, caso exista algum, está sendo lido?

Solução

Primeiro escreva o endereço em binário.

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
0	0	0	0	0	1	1	0	1	0	1	0	0	0	1	1

Você deve ser capaz de verificar que os níveis A_{15} a A_{10} ativam a saída $\overline{K1}$ do decodificador para selecionar o módulo-1 de RAM. Este módulo decodifica internamente as linhas de endereços A_9 até A_0 para selecionar a posição da qual o dado deve ser colocado no barramento de dados.

EXEMPLO 11-17

Qual módulo de RAM será escrito quando a CPU executar uma operação de escrita no endereço 1C65?

Solução

Escrevendo o endereço em binário, podemos constatar que $A_{12} = 1$. Isto produz um nível ALTO na saída da porta OR e na entrada C do decodificador. Com $A_{11} = A_{10} = 1$, as entradas do decodificador são 111, o que ativa a saída 7. As saídas $\overline{K0}$ até $\overline{K3}$ estarão inativas, e portanto nenhum dos módulos de RAM será habilitado. Em outras palavras, o dado colocado no barramento de dados pela CPU não será aceito por nenhuma das RAMs.

EXEMPLO 11-18

Determine a faixa de endereços para cada módulo na Fig. 11-48.

Solução

Cada módulo armazena 1.024 palavras de oito bits. Para determinar os endereços das palavras armazenadas em qualquer módulo, começamos determinando as condições do barramento de endereço que ativam a entrada de seleção (chip select) do módulo. Por exemplo, o módulo-3 será selecionado quando a saída do decodificador $\overline{K3}$ estiver em BAIXO (Fig. 11-49). $\overline{K3}$ fica em BAIXO para $CBA = 011$. Observando as linhas de endereço da CPU A_{15} até A_{10}, verificamos que o módulo-3 será habilitado quando o seguinte endereço for colocado no barramento de endereço:

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
0	0	0	0	1	1	x	x	x	x	x	x	x	x	x	x

Os x's abaixo de A_9 até A_0 indicam "don't care", pois essas linhas de endereço não são utilizadas pelo decodificador para selecionar o módulo-3. A_9 até A_0 pode ter qualquer combinação variando de 0000000000 até 1111111111, dependendo da palavra do módulo-3 que estiver sendo aces-

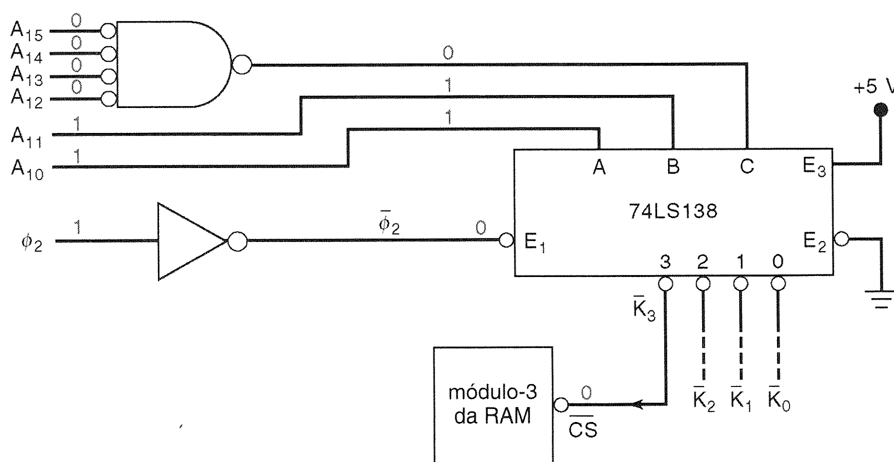


Fig. 11-49 O Exemplo 11-18 ilustra as condições do barramento de endereço necessárias para selecionar o módulo-3 da RAM.

sada. Desse modo, a faixa completa de endereços para o módulo-3 é determinada utilizando-se tudo 0 e depois tudo 1 para os *x*'s.

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	$\rightarrow 0C00_{16}$
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	$\rightarrow 0FFF_{16}$

Finalmente, isso nos fornece de 0C00 até 0FFF como a faixa de endereços hexadecimais do módulo-3. Quando a CPU coloca qualquer endereço nesta faixa no barramento de endereço, somente o módulo-3 ficará habilitado ou para uma leitura ou para uma escrita, dependendo do sinal $R/\overline{W}$.

Uma análise semelhante pode ser usada para determinar a faixa de endereços para cada um dos outros módulos de RAM. Os resultados são os seguintes:

- Módulo-0: 0000-03FF
- Módulo-1: 0400-07FF
- Módulo-2: 0800-0BFF
- Módulo-3: 0C00-0FFF

Repare que os quatro módulos se combinam para formar uma faixa de endereços total de 0000 até 0FFF.

Teste da Lógica de Decodificação

Em algumas situações, a parte do circuito da RAM relativa à lógica de decodificação pode ser testada utilizando-se as diversas técnicas que temos aplicado aos circuitos combinacionais. Ela pode ser testada aplicando-se sinais nas linhas mais significativas do endereço e ϕ_2 e monitorando-se as saídas do decodificador. Para fazer isso, deve ser possível desconectar facilmente a CPU dessas linhas de sinal. Se a CPU é um chip microprocessador em um soquete, ele pode ser simplesmente removido do seu soquete.

Uma vez desconectada a CPU, você pode fornecer os sinais A_{10}–A_{15} e ϕ_2 através de um circuito externo para realizar um teste estático, utilizando chaves operadas manualmente para cada sinal, ou um teste dinâmico, usando algum tipo de contador para ciclar através dos vários códigos de endereço. Com esses sinais de teste aplicados, as linhas de saída do decodificador podem ser verificadas. Técnicas comuns de rastreamento de sinais podem ser usadas para isolar qualquer falha na lógica de decodificação.

Se você não tiver acesso às linhas de endereço do sistema ou não tiver um modo conveniente para gerar os sinais lógicos estáticos, freqüentemente é possível forçar o sistema a gerar uma seqüência de endereços. A maioria dos sistemas computacionais que são utilizados para desenvolvimento possui um programa armazenado na ROM que permite ao usuário mostrar e alterar o conteúdo de qualquer posição de memória. Sempre que o computador acessar uma posição de memória, o endereço apropriado deve ser colocado no barramento, o que deve fazer com que a saída do decodificador vá para BAIXO, mesmo que seja por um tempo curto. Digite o seguinte comando no computador:

```
Display de 0400 a 07FF
```

Então coloque a ponta de prova lógica na saída $\overline{K1}$. A ponta de prova lógica deveria indicar a presença de pulsos durante o tempo em que os dados estivessem sendo apresentados na tela.

EXEMPLO 11-19

Um teste dinâmico é realizado na lógica de decodificação da Fig. 11-49, mantendo-se $\phi_2 = 1$ e conectando-se as saídas de um contador de seis bits nas entradas de endereço A_{10} até A_{15}. As saídas do decodificador são monitoradas enquanto o contador cicla repetidamente por todos os códigos de seis bits. Uma ponta de prova lógica verifica as saídas do decodificador e indica pulsos em $\overline{K1}$ e $\overline{K3}$, mas $\overline{K0}$ e $\overline{K2}$ permanecem em ALTO. Quais são as falhas mais prováveis?

Solução

É possível, mas bastante improvável, que $\overline{K0}$ e $\overline{K2}$ estejam ambas permanentemente em ALTO, devido a um curto-circuito interno ou externo com V_{CC}. Uma falha mais provável é uma ligação aberta entre A_{10} e a entrada *A* do decodificador, pois isso funcionaria como um nível lógico ALTO e impediria que qualquer saída par do decodificador fosse ativada. Também é possível que a entrada *A* do decodificador esteja em curto com V_{CC}, mas isso também não é razoável porque esse curto provavelmente afetaria a operação do contador que está fornecendo os endereços de entrada.

Teste do Sistema de RAM Completo

Os testes e a pesquisa de falhas na lógica de decodificação não vão revelar problemas com chips de memória e suas conexões nos barramentos da CPU. Os métodos mais comuns para testar a operação do sistema de RAM *completo* envolvem a escrita de padrões conhecidos de 0s e 1s em cada posição de memória e sua leitura para verificar que a posição armazenou corretamente o padrão. Enquanto diversos padrões diferentes podem ser utilizados, um dos mais amplamente usados é o "padrão xadrez". Neste padrão, 1s e 0s estão alternados como em 01010101. Uma vez que todas as posições tenham sido testadas usando-se esse padrão, o padrão é invertido (isto é, 10101010), e cada posição é testada novamente. Note que essa seqüência de testes verifica a habilidade de cada célula de armazenar e ler tanto 1 quanto 0. Como ele alterna 1s e 0s, o padrão xadrez também detecta qualquer interação ou curtos entre células adjacentes. Muitos outros padrões podem ser utilizados para detectar vários modos de falha dentro dos chips de RAM.

Nenhum teste de memória pode captar todas as possíveis falhas da RAM com 100 por cento de exatidão, embora possam mostrar que cada célula pode armazenar um 0 ou um 1. Algumas RAMs problemáticas podem ser sensíveis a um padrão. Por exemplo, uma RAM pode ser capaz de armazenar e fornecer 01010101 e 10101010, mas pode falhar quando 11100011 é armazenado. Mesmo para um sistema com pouca RAM, poderia levar um tempo proibitivamente longo para tentar armazenar e ler todos os padrões possíveis em cada posição. Por essa razão, se um sistema RAM passa pelo teste do padrão xadrez, você pode concluir que

ele *provavelmente* está bom; se ele falhar no teste, então *definitivamente* contém uma falha.

Testar manualmente milhares de posições de RAM armazenando e lendo padrões xadrez levaria centenas de horas, e obviamente não é factível. Testes com padrões na RAM usualmente são realizados automaticamente fazendo a CPU executar um programa de teste de memória ou conectando-se um instrumento especial de teste nos barramentos da RAM do sistema no lugar da CPU. De fato, em vários computadores e equipamentos baseados em microprocessadores, a CPU automaticamente roda um programa de teste de memória toda vez que é ligada; isto é chamado de **autoteste ao ligar** (*power-up self-test*). A rotina de autoteste (nós a denominaremos de AUTOTESTE) é armazenada em ROM e é executada sempre que o sistema é ligado ou quando o operador requisita isto pelo teclado. À medida que a CPU executa o AUTOTESTE, ela escreve padrões de teste e verifica a leitura em cada posição de RAM e apresenta algum tipo de mensagem para o usuário. Pode ser algo tão simples quanto um LED para indicar falha na memória ou pode ser uma mensagem mais esclarecedora enviada para a tela ou para a impressora. Mensagens típicas seriam:

```
RAM módulo-3 teste OK
TODA RAM operando adequadamente
Posição 027F falha nas posições dos bits 6 e 7
```

Com mensagens desse tipo e um conhecimento da operação da RAM do sistema, o responsável por resolver a falha pode determinar que ação é necessária para isolá-la. Os dois próximos exemplos ilustrarão o procedimento para o sistema com RAM de 4K × 8; por conveniência, seu diagrama esquemático está repetido na Fig. 11-50. Lembre-se, de nossos exemplos anteriores, de que os endereços para cada um dos módulos de RAM são os seguintes:

- Módulo-0: 0000-03FF
- Módulo-1: 0400-07FF
- Módulo-2: 0800-0BFF
- Módulo-3: 0C00-0FFF

EXEMPLO 11-20

As seguintes mensagens são enviadas para a tela após o AUTOTESTE rodar:

```
módulo-0 teste OK
módulo-1 teste OK
endereço 09A7 falha no bit 6
módulo-3 teste OK
```

Examine estas mensagens e decida o que fazer em seguida.

Solução

A posição que apresenta falha está no módulo-2. Não existe nenhuma falha de circuito externa ao módulo-2 que pudesse provocar apenas aquela única falha naquele endereço e nada nos outros. Desse modo, parece que existe uma falha interna no módulo-2. Entretanto, antes de chegarmos a esta conclusão, o programa de AUTOTESTE deve rodar muitas outras vezes para verificarmos se a mesma mensagem aparece. A mensagem original de falha pode ter sido o resultado de ruído em vez de uma célula de memória ruim; se for este o caso, outros testes não produzirão nenhuma mensagem de falha ou mensagens de falhas diferentes. Isso indicaria a necessidade de verificar fontes de ruído no sistema, tais como: capacitores de desacoplamento de alimentação inexistentes ou inoperantes, ou interferência entre linhas de sinal pouco espaçadas.

EXEMPLO 11-21

Considere a seguinte mensagem resultante do AUTOTESTE:

```
módulo-0 teste OK
endereço 0400 falha nos bits 0-7
endereço 0401 falha nos bits 0-7
endereço 0402 falha nos bits 0-7
   .        .     .    .    .
   .        .     .    .    .
endereço 07FE falha nos bits 0-7
endereço 07FF falha nos bits 0-7
módulo-2 teste OK
módulo-3 teste OK
```

Examine estas mensagens, relacione as falhas possíveis e descreva o que fazer em seguida.

Solução

Obviamente, todos os endereços no módulo-1 estão listados como tendo falhas, e portanto o módulo-1 não está funcionando. Diversas falhas possíveis poderiam estar provocando isto, tais como:

- Uma conexão interrompida na entrada $R/\overline{W}$ do módulo-1.
- Uma ligação aberta entre $\overline{K1}$ e o chip select do módulo-1.
- A saída $\overline{K1}$ do decodificador com problemas (fica em ALTO).
- A ligação $\overline{K1}/\overline{CS}$ externamente em curto com V_{CC}.
- Uma RAM ruim no módulo-1 (não responde ao $\overline{CS}$).
- V_{CC} ou terra desconectados no módulo-1.

As técnicas padronizadas de pesquisa de falhas podem ser usadas agora para isolar a falha.

Como esses exemplos mostraram, o programa de teste de memória é útil para restringir o problema a uma área específica do circuito. O responsável pela pesquisa de falhas utiliza a informação fornecida pelo teste de memória para determinar possíveis falhas e então prossegue para localizar a falha real.

Deve-se ressaltar que existem algumas falhas no circuito da RAM que impedem a CPU de executar o programa de AUTOTESTE na inicialização. Por exemplo, qualquer falha de circuito que faça uma linha de dados ou de endereço ficar permanentemente em BAIXO ou em ALTO causará erros de operação quando a CPU tentar executar o programa de AUTOTESTE a partir da ROM (lembre-se, a ROM compartilha os barramentos de endereço e de dados com a RAM). Quando isso acontece, você deve verificar as linhas dos barramentos, e, se alguma estiver permanentemente em BAIXO ou ALTO, utilize as técnicas padronizadas para determinar a causa exata desse problema.

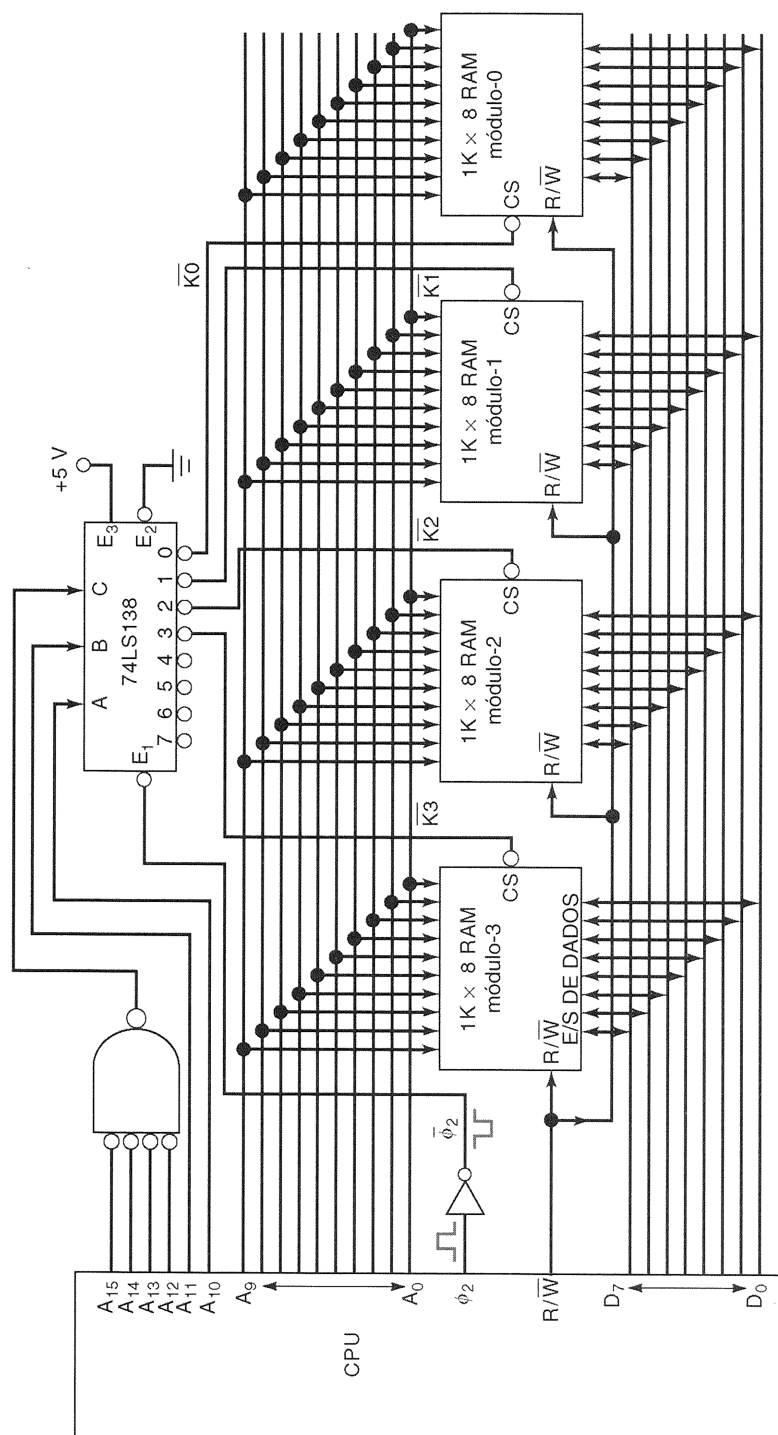


Fig. 11-50 Sistema com RAM de 4K × 8 repetido da Fig. 11-48.

Endereço	Dados
000	00000110
001	10010111
010	00110001
011	11111111
100	00000000
101	10000001
110	01000110
111	10010100 (Checksum)

(a)

Endereço	Dados
000	00000110
001	10010111
010	00110001
011	11111110 ← erro
100	00000000
101	10000001
110	01000110
111	10010100 (Checksum)

(b)

Fig. 11-51 O método do checksum para uma ROM de 8 × 8: (a) ROM com dados corretos; (b) ROM com erro nos seus dados.

Questões de Revisão

1. Qual é a função de ϕ_2 no circuito de RAM da Fig. 11-48?
2. O que é o teste xadrez? Por que ele é utilizado?
3. O que é o autoteste ao ligar?

11-22 TESTE DE ROM

Os circuitos de ROM num computador são muito semelhantes aos circuitos de RAM (compare as Fig. 11-45 e 11-48). A lógica de decodificação da ROM pode ser testada da mesma maneira descrita na seção anterior para a RAM do sistema. Entretanto, os chips de ROM devem ser testados de modo diferente dos chips de RAM, pois não podemos escrever padrões na ROM para lê-los de volta como fazíamos para a RAM. Diversos métodos são utilizados para verificar o conteúdo de um CI de ROM.

Num dos métodos, a ROM é colocada num soquete de um instrumento especial de teste que é tipicamente controlado por microprocessador. O instrumento especial de teste pode ser programado para ler todas as posições da ROM de teste e imprimir uma listagem do conteúdo de cada posição. A listagem pode então ser comparada com o que a ROM deveria conter. Com exceção dos chips de ROM de baixa capacidade, esse processo pode consumir muito tempo.

Em outro método mais eficiente, o instrumento de teste tem os dados corretos armazenados no seu próprio chip de *ROM de referência*. Então o instrumento de teste é programado para ler os conteúdos de todas as posições da ROM de teste e compará-los com os conteúdos da ROM de referência.

Um terceiro método utiliza um **checksum** (soma de verificação), que é um código especial colocado na última posição ou nas duas últimas posições do chip da ROM quando ela é programada. Esse código é calculado somando-se as palavras a serem armazenadas em todas as posições da ROM (excluindo aquelas que contêm o checksum). À medida que o instrumento de teste lê os dados de cada posição da ROM de teste, ele vai efetuando a soma destes dados e calcula seu próprio checksum. Então, compara o checksum calculado com aquele que está armazenado nas últimas posições da ROM, e eles devem ser iguais. Se for este o caso, existe uma alta probabilidade de a ROM estar boa (existe uma pequena chance de que uma combinação de erros nos dados da ROM de teste possa produzir o mesmo valor de checksum). Se eles não forem iguais, então certamente existe um problema na ROM de teste.

A idéia do checksum está ilustrada na Fig. 11-51(a) para uma ROM muito pequena. A palavra armazenada no último endereço é a soma de oito bits das outras sete palavras (ignorando-se o carry do MSB). Quando esta ROM é programada, o checksum é colocado na última posição. A Fig. 11-51(b) mostra os dados que foram lidos de uma ROM ruim que foi originalmente programada com os dados de (a). Note o erro na palavra do endereço 011. À medida que o instrumento de teste lê o dado de cada posição da ROM, ele calcula seu próprio checksum para aqueles dados. Por causa do erro, o checksum calculado será 10010011. Quando o instrumento de teste comparar isto com o valor do checksum armazenado na posição 111 da ROM, ele verifica a diferença e indica um erro na ROM. Naturalmente, a posição exata do erro não pode ser determinada.

O método do checksum também pode ser utilizado por um computador ou equipamento baseado em microprocessador durante o autoteste ao ligar para verificar o conteúdo das ROMs do sistema. Novamente, como no autoteste usado para RAM, a CPU deve executar um programa ao ser ligada que realiza o teste de checksum em cada chip ROM e enviar algum tipo de mensagem com os resultados. O programa de autoteste em si deve ser localizado na ROM, e portanto qualquer erro nessa ROM pode impedir a execução bem-sucedida dos testes de checksum.

Questões de Revisão

1. O que é checksum? Qual é o seu propósito?

RESUMO

1. Todos os dispositivos de memória armazenam níveis lógicos (1s e 0s) numa estrutura do tipo matriz. O tamanho de cada palavra binária (número de bits) que é armazenada varia de acordo com o dispositivo de memória. Estes valores binários são referenciados como *dados*.
2. O local (posição) no dispositivo de memória onde o dado é armazenado é identificado por outro número binário denominado *endereço*. Cada posição de memória tem um único endereço.
3. Todos os dispositivos de memória operam do mesmo modo em geral. Para escrever dados na memória, o endereço a ser acessado é colocado nas entradas de endereço, o dado a ser armazenado é aplicado nas entradas de dados e os sinais de controle são manipulados para armazenar o dado. Para ler dados na memória, o endereço é aplicado, os sinais de controle são manipulados e o dado aparece nos pinos de saída.
4. Dispositivos de memória são freqüentemente utilizados com um microprocessador que gera os sinais de endereço e de controle e fornece os dados a serem armazenados ou utiliza os dados da memória. Leitura e escrita *sempre* são feitas sob o ponto de vista da CPU. A escrita coloca dados na memória, e a leitura recupera os dados da memória.
5. A maioria das memórias somente de leitura (ROMs) tem seus dados colocados uma vez, e a partir daí seu conteúdo não muda. Esse processo de armazenamento é chamado *programação*. Elas não perdem seus dados quando a energia é removida do dispositivo. As MROMs são programadas durante o processo de fabricação. As PROMs são programadas uma vez pelo usuário. As EPROMs são como as PROMs, mas podem ser apagadas utilizando-se luz UV. As EEPROMs e as memórias flash são dispositivos eletricamente apagáveis e podem ter seu conteúdo alterado após a programação. CD ROMs são usados para armazenamento de massa de informações que não necessitam ser alteradas.
6. Dispositivos de lógica programável (PLDs) são dispositivos que contêm um arranjo de elementos de circuitos lógicos básicos, tais como portas AND, OR e INVERSOR, como também flip-flops e buffers tristate. Uma seção de memória também está contida em cada PLD. Os elementos lógicos podem ser interconectados para formar circuitos lógicos. Cada interconexão é controlada por uma determinada posição de bit na memória. PLDs podem implementar qualquer função lógica combinacional, limitada apenas pelo número de entradas lógicas e pelo número de saídas em potencial para um determinado dispositivo programável. Circuitos seqüenciais são limitados pelo número de flip-flops disponíveis e pelas restrições de arquitetura para acioná-los pelo clock.
7. PLDs são programados utilizando programas em um PC e controlando um programador que pode ser configurado para aceitar o componente que você está usando. Com sua versatilidade e facilidade de uso, os PLDs estão gradualmente substituindo chips de lógica convencional na implementação de sistemas digitais.
8. Memória de acesso aleatório (RAM) é um termo genérico dado aos dispositivos que podem ter dados facilmente escritos e lidos. Os dados são retidos num dispositivo RAM somente enquanto a energia estiver aplicada.
9. RAM estática (SRAM) utiliza elementos de armazenamento que são basicamente circuitos do tipo latch. Uma vez que os dados são armazenados, eles permanecerão inalterados enquanto a energia estiver aplicada ao chip. A RAM estática é mais fácil de usar, mas é mais cara por bit e consome mais energia do que a RAM dinâmica.
10. RAM dinâmica (DRAM) usa capacitores para armazenar dados através da carga e descarga dos mesmos. A simplicidade da célula de armazenamento permite às DRAMs armazenarem uma grande quantidade de dados. Como a carga dos capacitores deve ser restaurada regularmente, as DRAMs são mais complicadas de usar do que as SRAMs. Circuitos extras são adicionados freqüentemente aos sistemas com DRAMs para controlar a leitura, a escrita e os ciclos de refresh. Em muitos dispositivos mais novos, essas características estão sendo integradas no próprio chip DRAM. O objetivo da tecnologia DRAM é colocar mais bits num pedaço menor de silício, de modo que ela consuma menos energia e responda mais rapidamente.
11. Sistemas de memória requisitam uma ampla faixa de configurações diferentes. Chips de memória podem ser combinados para implementar qualquer configuração desejada, no caso de seu sistema necessitar de mais bits por posição ou maior capacidade total de palavras. Todos os diversos tipos de ROM e RAM podem ser combinados dentro de um mesmo sistema de memória.

TERMOS IMPORTANTES

memória principal
memória auxiliar
célula de memória
palavra de memória
byte
capacidade
densidade
endereço
operação de leitura
operação de escrita
tempo de acesso
memória volátil
RAM
SAM
RWM
ROM
RAM estática (SRAM)
RAM dinâmica (DRAM)
barramento de endereço
barramento de dados
barramento de controle
programador-programa-programação
chip select
power-down
fusível
EEPROM
memória flash
firmware
programa bootstrap
dispositivo de lógica programável (PLD)
arranjo lógico programável (PAL)
polaridade de saída programável
arranjo lógico programável em campo (FPLA)
PLDs complexos
arranjo de portas programáveis no campo (FPGA)
circuito integrado de aplicação específica (ASIC)
refresh
JEDEC
multiplexação de endereços
strobe do endereço da linha (*RAS*)
strobe do endereço da coluna (*CAS*)
refresh com apenas $\overline{RAS}$
contador de refresh
controlador DRAM
memória refletida
mapa de memória
FIFO

buffer de transferência de dados
autoteste ao ligar
checksum

PROBLEMAS

SEÇÕES 11-1 A 11-3

11-1. Uma determinada memória tem capacidade de 16K × 32. Quantas palavras ela armazena? Qual é o número de bits por palavra? Quantas células de memória ela possui?

11-2. Quantas linhas de endereço são necessárias para a memória do Problema 11-1?

11-3. Qual é a capacidade de uma memória que tem 16 entradas de endereço, quatro entradas de dados e quatro saídas de dados?

11-4. Certa memória armazena 8K palavras de 16 bits. Quantas linhas de dados de entrada e linhas de dados de saída ela possui? Quantas linhas de endereço ela tem? Qual é a sua capacidade em bytes?

QUESTÕES DE FIXAÇÃO

11-5. Defina cada um dos seguintes termos.
(a) RAM
(b) RWM
(c) ROM
(d) Memória interna
(e) Memória auxiliar
(f) Capacidade
(g) Volátil
(h) Densidade
(i) Leitura
(j) Escrita

11-6. **(a)** Quais são os três barramentos no sistema de memória de um computador?
(b) Qual barramento é utilizado pela CPU para selecionar a posição de memória?
(c) Qual barramento é utilizado para levar os dados da memória para a CPU durante uma operação de leitura?
(d) Qual é a fonte de dados para o barramento de dados durante uma operação de escrita?

SEÇÕES 11-4 E 11-5

11-7. Consulte a Fig. 11-6. Determine as saídas de dados para cada uma das seguintes condições de entrada:
(a) $[A] = 1011$; $CS = 1$
(b) $[A] = 0111$; $CS = 0$

11-8. Consulte a Fig. 11-7.
(a) Que registrador é habilitado pelo endereço de entrada 1011?
(b) Que código de endereço de entrada seleciona o registrador 4?

11-9. Uma certa ROM tem capacidade de 16K × 4 e uma estrutura interna análoga àquela mostrada na Fig. 11-7.
(a) Quantos registradores existem na matriz?
(b) Quantos bits existem por registrador?
(c) De que decodificadores ela necessita?

QUESTÃO DE FIXAÇÃO

11-10. **(a)** *Verdadeiro ou falso*: ROMs não podem ser apagadas.
(b) O que significa *programar* ou *"queimar"* uma ROM?
(c) Defina tempo de acesso de uma ROM.
(d) Quantas entradas de dados, saídas de dados e entradas de endereço são necessárias para uma ROM de 1.024 × 4?
(e) Qual é a função dos decodificadores num chip de ROM?

SEÇÃO 11-6

C, D
11-11. A Fig. 11-52 mostra como os dados de uma ROM podem ser transferidos para um registrador externo. A ROM tem os seguintes parâmetros de temporização: $t_{ACC} = 250$ ns e $t_{OE} = 120$ ns. Admita que novas entradas de endereço foram aplicadas na ROM 500 ns antes da ocorrência do pulso TRANSFER. Determine a duração mínima do pulso TRANSFER para que a transferência de dados seja confiável.

C, D
11-12. Repita o Problema 11-11 para o caso em que as entradas de endereço são alteradas 70 ns antes do pulso TRANSFER.

SEÇÕES 11-7 E 11-8

11-13. QUESTÃO DE FIXAÇÃO
Para cada sentença a seguir, indique o tipo de memória sendo descrito: MROM, PROM, EPROM, EEPROM ou flash. Algumas sentenças correspondem a mais de um tipo de memória.
(a) Pode ser programada pelo usuário, mas não pode ser apagada.
(b) É programada pelo fabricante.

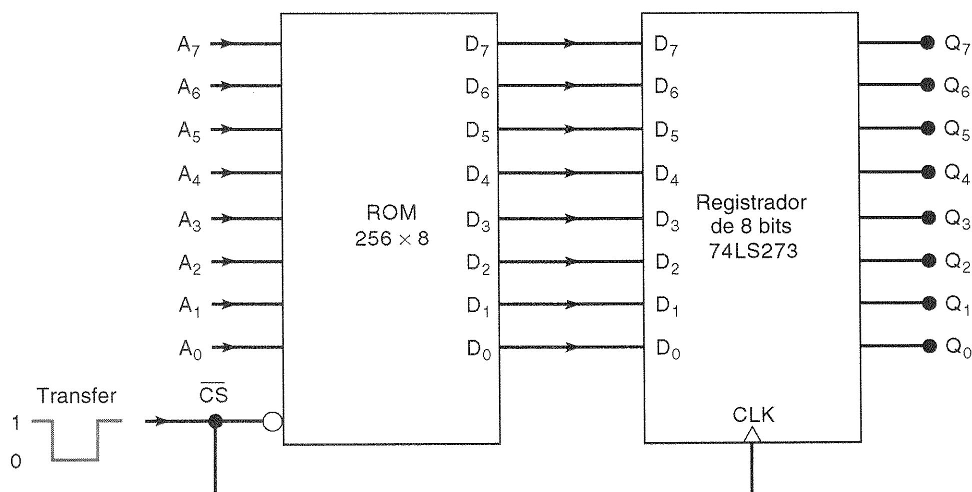


Fig. 11-52 Problema 11-11.

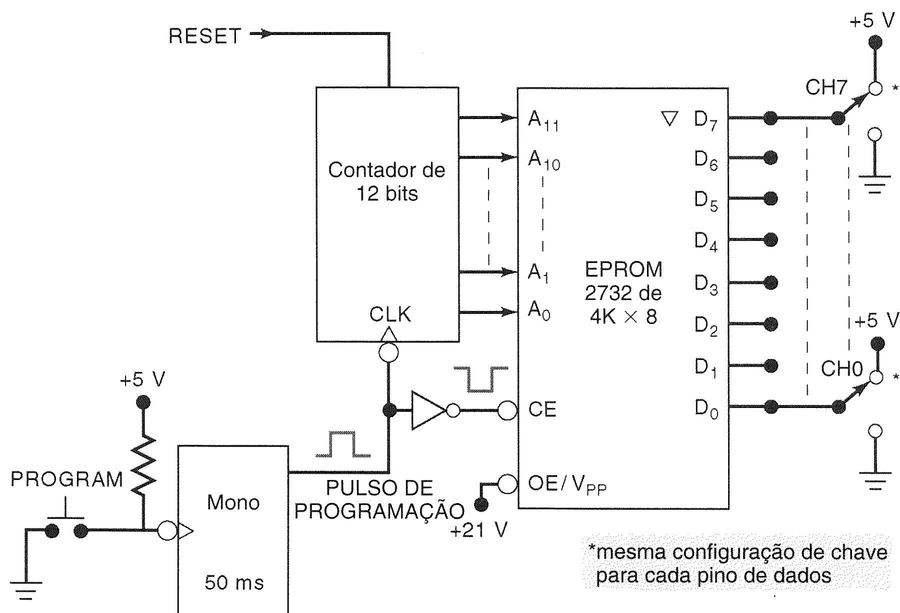


Fig. 11-53 Problema 11-16.

- **(c)** É volátil.
- **(d)** Pode ser apagada e reprogramada diversas vezes.
- **(e)** Palavras individuais podem ser apagadas e reescritas.
- **(f)** É apagada com luz UV.
- **(g)** É apagada eletricamente.
- **(h)** Utiliza conexões com fusíveis.
- **(i)** Pode ser toda apagada ou em setores de 512 bytes.
- **(j)** Não tem que ser retirada do sistema para ser apagada e reprogramada.
- **(k)** Necessita de uma fonte de alimentação especial para reprogramação.
- **(l)** O tempo de apagamento é aproxidamente de 15 a 20 minutos.

11-14. Quais transistores na Fig. 11-9 estarão conduzindo quando $A_1 = A_0 = 1$ e $\overline{EN} = 0$?

11-15. Altere as conexões da MROM na Fig. 11-9 de modo que ela armazene a função $y = 3x + 5$.

D

11-16. A Fig. 11-53 mostra um circuito simples para programar manualmente uma EPROM 2732. Cada pino de dados da EPROM está conectado a uma chave que pode colocar o nível em 0 ou em 1. As entradas de endereço são acionadas por um contador de 12 bits. O pulso de programação de 50 ms é gerado por um monoestável cada vez que o botão PROGRAM é acionado.

- **(a)** Explique como esse circuito pode ser usado para programar seqüencialmente as posições da EPROM com os dados desejados.
- **(b)** Mostre como os chips 74293s e 74121 podem ser utilizados para implementar esse circuito.
- **(c)** A trepidação de contato teria algum efeito sobre a operação do circuito?

N

11-17. A Fig. 11-54 mostra um chip de memória flash 28F256A conectado a uma CPU através do barramento de dados e do barramento de endereço. A CPU pode escrever ou ler da memória flash enviando o endereço de memória desejado e gerando os sinais de controle apropriados para o chip [Fig. 11-15(b)]. A CPU também pode escrever no registrador de comandos do chip (Fig. 11-16), gerando os sinais de controle apropriados e enviando o código do comando desejado pelo barramento de dados. Para esta última operação, a CPU não tem que enviar um endereço de memória específico para o chip; em outras palavras, as linhas de endereços são "don't cares".

- **(a)** Considere a seguinte seqüência de operações da CPU. Determine o que acontecerá com a memória flash quando a seqüência for concluída. Suponha que o registrador de comandos tenha 00_{16}.
  1. A CPU coloca 20_{16} no barramento de dados e pulsa $\overline{CE}$ e $\overline{WE}$ em BAIXO enquanto mantém $\overline{OE}$ em ALTO. O barramento de endereço está com 0000_{16}.
  2. A CPU repete o passo 1.
- **(b)** Depois de isso ter sido executado, a CPU executa a seqüência a seguir. Determine o que isso faz com o chip de memória flash.
  1. A CPU coloca 40_{16} no barramento de dados e pulsa $\overline{CE}$ e $\overline{WE}$ em BAIXO enquanto mantém $\overline{OE}$ em ALTO. O barramento de endereço está com 0000_{16}.
  2. A CPU coloca $3C_{16}$ no barramento de dados e 2300_{16} no barramento de endereço e pulsa $\overline{CE}$ e $\overline{WE}$ em BAIXO enquanto mantém $\overline{OE}$ em ALTO.

SEÇÃO 11-9

N

11-18. Uma outra aplicação de ROMs é a geração de sinais de controle e temporização. A Fig. 11-55 mostra uma ROM de 16 × 8 com suas entradas de endereço acionadas por um contador de módulo 16, de modo que os endereços da ROM são incrementados com cada pulso de entrada. Suponha que a ROM é programada como na Fig. 11-6 e esboce as formas de onda de cada uma das saídas da ROM conforme os pulsos vão sendo aplicados. Ignore os tempos de atraso da ROM. Considere que o contador começa em 0000.

D

11-19. Altere o programa armazenado na ROM do Problema 11-18 para gerar a forma de onda D_7 da Fig. 11-56.

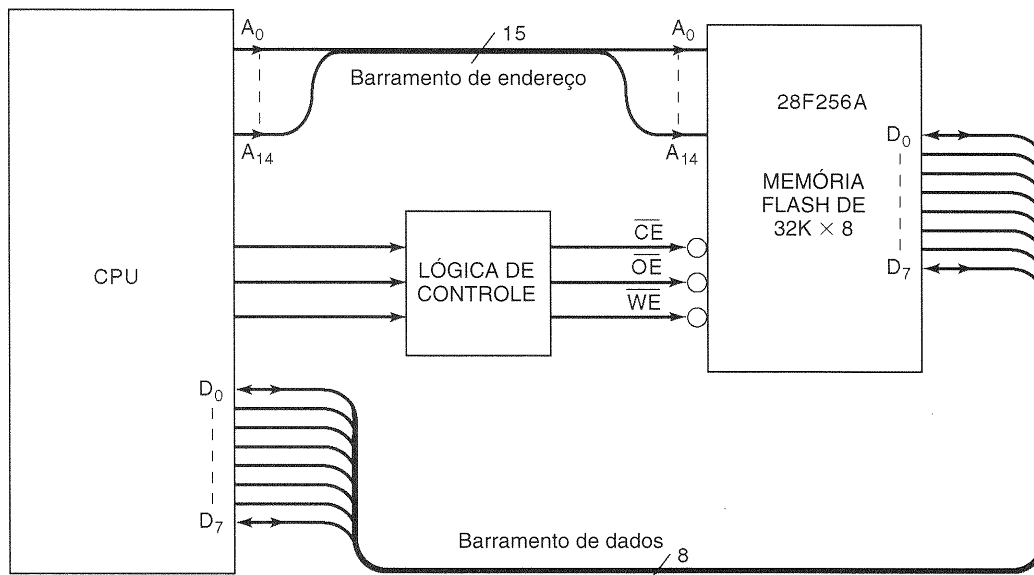


Fig. 11-54 Problema 11-17.

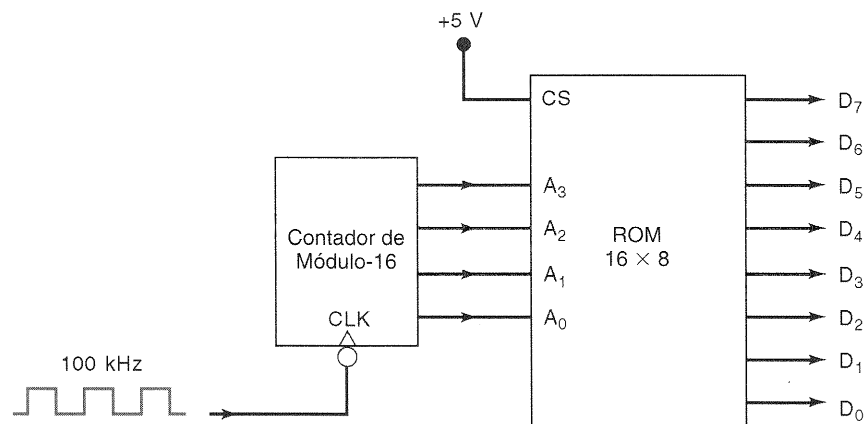


Fig. 11-55 Problema 11-18.

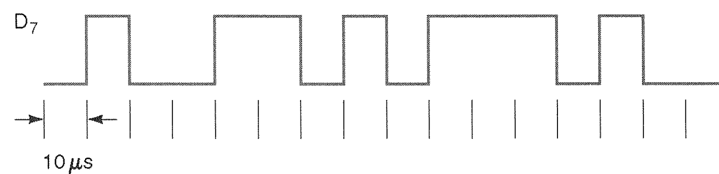
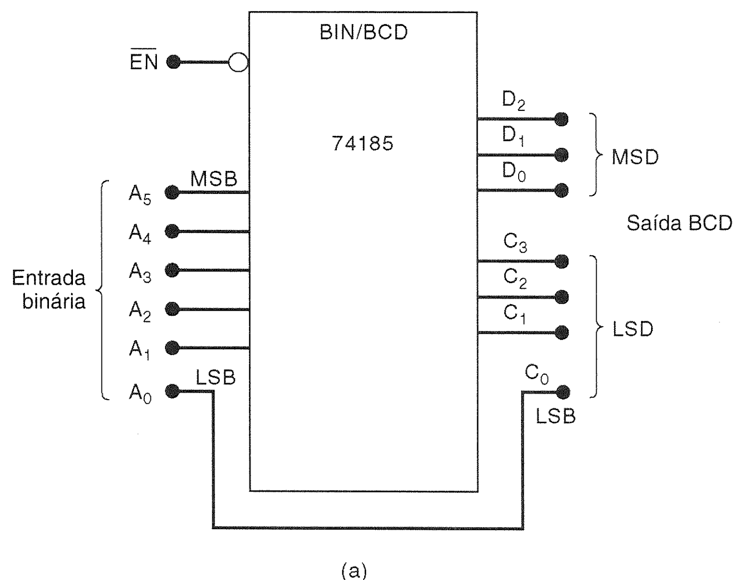


Fig. 11-56 Problema 11-19.



Binário						BCD						
A_5	A_4	A_3	A_2	A_1	A_0	D_2	D_1	D_0	C_3	C_2	C_1	C_0
1	0	0	1	0	1	0	1	1	0	1	1	1
0	0	1	1	1	0	0	0	1	0	1	0	0
1	1	1	1	1	0	1	1	0	0	0	1	0
0	1	1	1	0	0	0	1	0	1	0	0	0
1	1	0	1	0	1							
0	0	0	1	1	0							
						0	0	1	1	0	0	1
						1	0	0	1	0	0	0

(b)

Fig. 11-57 Problema 11-20.

C, N

11-20. A Fig. 11-57(a) mostra o símbolo lógico para o CI 74185, que é uma ROM de "prateleira" que é programada como um *conversor binário-BCD*. Ela converte uma entrada binária de seis bits em uma saída BCD de dois dígitos. Note que o LSB da entrada binária de seis bits não está conectado na ROM como uma das entradas de endereço; em vez disso, ele é simplesmente levado para a saída e se torna o LSB do LSD da saída. A Fig. 11-57(b) mostra parte da tabela-verdade ilustrando o conteúdo de algumas posições nesta ROM. Examine-a para verificar como a conversão de binário para BCD está sendo feita. Depois preencha as entradas que faltam na tabela.

D

11-21. Consulte o gerador de funções da Fig. 11-17.

(a) Que freqüência de clock resulta numa onda senoidal de 100 Hz na saída?

(b) Que método poderia ser usado para variar a amplitude de pico a pico da onda senoidal?

C

11-22. Para o ML2035 da Fig. 11-18, considere que o valor 038E (hexa) no latch produzirá a freqüência desejada. Desenhe o diagrama de tempo para as entradas $\overline{LATI}$, *SID* e *SCK*, admitindo que o LSB é deslocado primeiro.

SEÇÃO 11-12

11-23. (a) Desenhe o símbolo lógico para um MCM101514, uma RAM estática CMOS com organização de 256K × 4, entradas e saídas de dados separadas e um *chip enable* ativo em BAIXO.

(b) Desenhe o símbolo lógico para um MCM6249, uma RAM estática CMOS com organização de 1M × 4, entradas e saídas em comum, habilitação de seleção em nível BAIXO e uma habilitação de saída ativa em BAIXO.

SEÇÃO 11-13

11-24. Uma determinada RAM estática tem os seguintes parâmetros de temporização (em nanossegundos):

t_{RC} = 100 t_{AS} = 20
t_{ACC} = 100 t_{AH} = não fornecido
t_{CO} = 70 t_W = 40
t_{OD} = 30 t_{DS} = 10
t_{WC} = 100 t_{DH} = 20

(a) Durante um ciclo de leitura, quanto tempo após as linhas de endereço estarem estáveis dados válidos vão aparecer nas saídas?

(b) Quanto tempo as saídas de dados vão permanecer válidas após $\overline{CS}$ retornar para ALTO?

(c) Quantas operações de leitura podem ser realizadas por segundo?

(d) Durante um ciclo de escrita, por quanto tempo $R/\overline{W}$ e $\overline{CS}$ devem ser mantidos em ALTO após as linhas de endereço estarem estáveis?

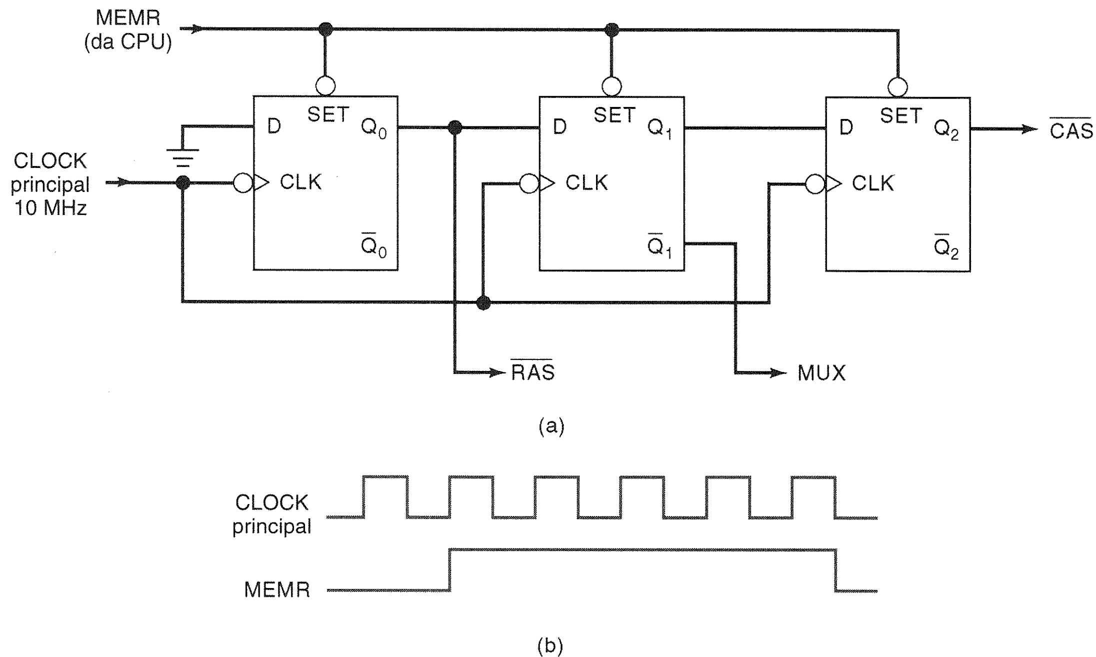


Fig. 11-58 Problema 11-26.

(e) Qual é o tempo mínimo que um dado de entrada deve permanecer válido para uma operação de escrita confiável acontecer?

(f) Por quanto tempo as entradas de endereço devem permanecer estáveis após $R/\overline{W}$ e $\overline{CS}$ retornarem a ALTO?

(g) Quantas operações de escrita podem ser realizadas por segundo?

SEÇÕES 11-14 A 11-18

11-25. Desenhe o símbolo lógico para o TMS4256, que é uma DRAM de 256K × 1. Quantos pinos são economizados pela utilização de multiplexação para esta DRAM?

D

11-26. A Fig. 11-58(a) mostra um circuito que gera os sinais de $\overline{RAS}$, $\overline{CAS}$ e MUX necessários para a operação apropriada do circuito da Fig. 11-34(b). O clock principal de 10 MHz fornece a temporização básica para o computador. O sinal de pedido de memória (*MEMR— memory request*) é gerado pela CPU, em sincronismo com o clock, conforme mostrado na parte (b) da figura. *MEMR* está normalmente em BAIXO e é ativado em ALTO sempre que a CPU deseja acessar a memória para uma operação de leitura ou de escrita. Determine as formas de onda em Q_0, $\overline{Q}_1$ e Q_2 e compare-as com as formas de onda desejadas da Fig. 11-35.

D

11-27. Mostre como conectar dois multiplexadores 74157 (Seção 9-7) para fornecer a função de multiplexação requisitada na Fig. 11-34(b).

11-28. Consulte os sinais na Fig. 11-36. Descreva o que ocorre em cada um dos instantes de tempo identificados.

11-29. Repita o Problema 11-28 para a Fig. 11-37.

C

11-30. A 21256 é uma DRAM de 256K × 1 constituída por uma matriz de células de 512 × 512. As células devem ser refrescadas dentro de 4 ms para que os dados sejam mantidos. A Fig. 11-39(a) mostra os sinais usados para executar um ciclo de refresh do tipo $\overline{CAS}$ antes de $\overline{RAS}$. Sempre que um ciclo desses ocorre, os circuitos de refresh internos ao chip refrescam uma linha da matriz no endereço de linha especificado por um contador de refresh. O contador é incrementado após cada refresh. A que taxa os ciclos de $\overline{CAS}$ antes de $\overline{RAS}$ deveriam ser aplicados de modo que todos os dados fossem mantidos?

11-31. Estude o diagrama de blocos funcionais da DRAM TMS44100 na Fig. 11-33.

(a) Quais são as dimensões reais da matriz de células?

(b) Se a matriz de células realmente fosse quadrada, quantas linhas deveriam existir?

(c) Como isso afetaria o tempo de refresh?

SEÇÃO 11-19

D

11-32. Mostre como combinar dois chips RAM 6206 (Fig. 11-26) para produzir um módulo de 32K × 16.

D

11-33. Mostre como conectar dois dos chips RAM 6264 simbolizados na Fig. 11-29 para produzir um módulo de 16K × 8. O circuito não necessita de nenhuma lógica adicional. Desenhe um mapa de memória mostrando a faixa de endereços para cada chip de RAM.

D

11-34. Descreva como modificar o circuito da Fig. 11-43 de modo que ele possua uma capacidade total de 16K × 8. Utilize os mesmos tipos de chips de PROM.

D

11-35. Modifique o circuito de decodificação da Fig. 11-43 para operar num barramento de 16 linhas de endereço (isto é, inclua A_{13}, A_{14} e A_{15}). As quatro PROMs devem permanecer nas mesmas faixas de endereços hexadecimais.

C

11-36. Para o sistema de memória da Fig. 11-44, admita que a CPU está armazenando um byte de dado no endereço 4000 (hexa) do sistema.

(a) Em qual dos chips o byte é armazenado?

(b) Existe algum outro endereço nesse sistema que possa acessar esse byte?

(c) Responda aos itens (a) e (b) supondo que a CPU tenha armazenado um byte no endereço 6007. (*Sugestão:* Lembre-se de que a EEPROM não está completamente decodificada.)

(d) Admita que o programa esteja armazenando uma seqüência de bytes de dados na EEPROM e que acabou de completar o byte 2.048 no endereço 67FF. Se o programador permitir que ele armazene um byte a mais no endereço 6800, qual será o efeito nos primeiros 2.048 bytes?

D

11-37. Desenhe o diagrama completo para uma memória de 256K × 8 que utiliza chips RAM com as seguintes especificações: capacidade de 64K × 4, linhas de entrada e saída comuns e duas entradas de seleção do chip ativas em BAIXO. [*Sugestão:* O circuito pode ser projetado utilizando-se apenas dois inversores (mais os chips de memória).]

SEÇÃO 11-21

11-38. Modifique o circuito com RAM da Fig. 11-48 do seguinte modo: substitua a porta OR por uma porta AND e desconecte sua saída de C; conecte a saída da AND em E_3; conecte C na terra. Determine a faixa de endereços para cada módulo RAM.

C, D

11-39. Mostre como expandir o sistema da Fig. 11-48 para 8K × 8 com endereços variando de 0000 até 1FFF. (*Sugestão:* Isto pode ser feito adicionando-se os módulos de memória necessários e modificando a lógica de decodificação existente.)

T

11-40. Um teste dinâmico é realizado na lógica de decodificação da Fig. 11-48 mantendo-se $\phi_2 = 1$ e conectando-se as saídas de um contador de seis bits nas entradas de endereço A_{10} até A_{15}. As saídas do decodificador são monitoradas com um osciloscópio (ou com um analisador lógico) conforme o contador é continuamente acionado pelo clock de 1 MHz. A Fig. 11-59(a) mostra os sinais apresentados. Quais são as falhas mais prováveis?

C, T

11-41. Repita o Problema 11-40 para as saídas do decodificador mostradas na Fig. 11-59(b).

C, D

11-42. Considere o sistema RAM da Fig. 11-48. O teste do padrão xadrez não será capaz de detectar certos tipos de falhas. Por exemplo, suponha que existe uma interrupção na conexão para a entrada A do decodificador. Se um AUTOTESTE com o padrão xadrez for realizado nesse circuito, as mensagens apresentadas vão afirmar que a memória está OK.

(a) Explique por que a falha do circuito não foi detectada.

(b) Como você modificaria o AUTOTESTE para que falhas como essas fossem detectadas?

T

11-43. Suponha que os módulos de 1K × 8 utilizados na Fig. 11-48 são formados por dois chips RAM de 1K × 4. As mensagens a seguir são exibidas quando o autoteste ao ligar é realizado neste sistema de RAM:

Fig. 11-59 Problemas 11-40 e 11-41.

```
módulo-0 teste OK
módulo-1 teste OK
endereço 0800 falha nos bits 4-7
endereço 0801 falha nos bits 4-7
endereço 0802 falha nos bits 4-7
   .   .   .   .   .   .
   .   .   .   .   .   .
   .   .   .   .   .   .
endereço 0BFE falha nos bits 4-7
endereço 0BFF falha nos bits 4-7
módulo-3 teste OK
```

Examine estas mensagens e relacione as possíveis falhas.

T

11-44. As mensagens seguintes foram apresentadas quando o autoteste ao ligar foi realizado no sistema RAM da Fig. 11-50.

```
módulo-0 teste OK
módulo-1 teste OK
módulo-2 teste OK
endereço 0C00 falha no bit 7
endereço 0C01 falha no bit 7
endereço 0C02 falha no bit 7
   .       .  .    .   .
   .       .  .    .   .
   .       .  .    .   .
endereço 0FFE falha no bit 7
endereço 0FFF falha no bit 7
```

Examine estas mensagens e relacione as possíveis falhas.

T

11-45. Que mensagens seriam exibidas quando o autoteste ao ligar fosse realizado no sistema RAM da Fig. 11-50, se houvesse um curto entre as saídas $\overline{K2}$ e $\overline{K3}$ do decodificador?

SEÇÃO 11-22

T

11-46. Considere a ROM de 16 × 8 na Fig. 11-6. Substitua o dado armazenado na posição cujo endereço é 1111 com um checksum calculado a partir da soma das outras 15 palavras.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 11-1

1. Vide texto.
2. 16 bits por palavra; 8.192 palavras; 131.072 bits ou células
3. Numa operação de leitura, uma palavra é copiada de uma posição de memória e é transferida para outro dispositivo. Numa operação de escrita, uma nova palavra é colocada na posição de memória, substituindo aquela palavra previamente armazenada ali.
4. Verdadeiro
5. SAM: o tempo de acesso não é constante mas depende da posição física da palavra que está sendo acessada. RAM: o tempo de acesso é o mesmo para qualquer posição de memória.
6. RWM é uma memória que pode ser lida e escrita com a mesma facilidade. ROM é uma memória que é principalmente lida e é raramente escrita.
7. Falso; seus dados devem ser refrescados periodicamente.

SEÇÃO 11-2

1. 14, 12, 12
2. Comanda a memória a realizar uma operação de leitura ou uma operação de escrita.
3. Quando está no seu estado ativo, essa entrada habilita a memória a realizar a operação de leitura ou de escrita selecionada pela entrada $R/\overline{W}$. Quando está no seu estado inativo, essa entrada desabilita a memória, de modo que ela não pode realizar a função de leitura nem a de escrita.

SEÇÃO 11-3

1. Linhas de endereço, linhas de dados, linhas de controle
2, 3. Vide texto.

SEÇÃO 11-4

1. Verdadeiro
2. Aplicação das entradas de endereço desejadas; ativação da(s) entrada(s) de controle; dados aparecem nas saídas de dados.
3. Procedimento de colocação de dados na ROM

SEÇÃO 11-5

1. $A_3A_2A_1A_0$ = 1001
2. O decodificador de seleção de linhas ativa uma das entradas de habilitação de todos os registradores na linha selecionada. O decodificador de seleção de colunas ativa uma das entradas de habilitação de todos os registradores na coluna selecionada. O buffer de saída passa o dado do barramento de dados interno para os pinos de saída da ROM quando a entrada *CS* é ativada.

SEÇÃO 11-7

1. Falso; pelo fabricante
2. Uma PROM pode ser programada uma vez pelo usuário. Ela não pode ser apagada nem reprogramada.
3. Verdadeiro
4. Pela exposição à luz UV
5. Verdadeiro
6. Automaticamente programa os dados nas células de memória um endereço de cada vez
7. Uma EEPROM pode ser eletricamente apagada e reprogramada sem ser retirada do circuito e num tempo muito menor do que uma EPROM.
8. Baixa densidade; alto custo
9. EEPROM
10. Um

SEÇÃO 11-8

1. Eletricamente apagável e programável no circuito
2. Densidade mais alta; custo mais baixo
3. Tempos curtos de apagamento e programação
4. Para as operações de apagamento e programação
5. O conteúdo desse registrador controla todas as funções internas do chip
6. Para confirmar que uma posição de memória foi apagada com sucesso (isto é, todos os bits de dados iguais a 1)
7. Para confirmar que uma posição de memória foi programada com o dado correto

SEÇÃO 11-9

1. Programas de microcomputador armazenados em ROM
2. Na energização do sistema, o computador executa um pequeno programa na ROM para inicializar o hardware do sistema e carregar o sistema operacional da memória de massa (disco).
3. Circuito que recebe um dado representado em um tipo de código e o converte para outro tipo de código.
4. Contador, ROM, conversor D/A, filtro passa-baixa
5. Elas são não-voláteis, rápidas, confiáveis, pequenas, e consomem pouco.

SEÇÃO 11-10

1. Um CI que contém um grande número de portas, FFs e registradores cujas interconexões podem ser modificadas pelo usuário para realizar funções específicas

2. $O_1 = A$
3. Um fusível intacto
4. Uma ligação permanente
6. Uma PAL tem um arranjo AND programável; a PROM tem um arranjo OR programável.
8. Numa FPLA, ambos os arranjos AND e OR são programáveis.
9. Vide a Fig. 11-24.
10. Para inverter ou não inverter as saídas do dispositivo
11. Pode ser apagada e reprogramada
12. Um FPGA

SEÇÃO 11-12
1. Os endereços desejados aplicados nas entradas de endereço; $R/\overline{W} = 1$; *CS* ou *CE* ativado
2. Para reduzir o número de pinos
3. 24, incluindo V_{CC} e terra.

SEÇÃO 11-13
1. Células de SRAMs são flip-flops; células de DRAM usam capacitores.
2. CMOS
3. RAM
4. CPU
5. Tempos de ciclo de leitura e escrita
6. Falso; quando $\overline{WE}$ está BAIXO, os pinos de entrada e saída funcionam como entradas de dados, não importando o estado de $\overline{OE}$ (segunda entrada da tabela de modo).
7. A_{13} pode permanecer conectado ao pino 26. A_{14} deve ser retirado, e o pino 27 deve ser ligado a +5 V.

SEÇÃO 11-14
1. Geralmente de menor velocidade; necessita ser refrescada
2. Baixa potência; alta capacidade; menor custo por bit
3. DRAM

SEÇÃO 11-15
1. 256 linhas × 256 colunas
2. Economiza pinos no chip.
3. 1M = 1.024K = 1.024 × 1.024. Logo, existem 1.024 linhas por 1.024 colunas. Como 1.024 = 2^{10}, o chip necessita de 10 entradas de endereço.
4. $\overline{RAS}$ é utilizado para armazenar o endereço da linha no registrador de endereço de linha da DRAM. $\overline{CAS}$ é utilizado para armazenar o endereço da coluna no registrador de endereço de coluna.
5. *MUX* multiplexa o endereço completo em endereços de linha e de coluna para entrada na DRAM.

SEÇÃO 11-16
1. (a) Verdadeiro
(b) Falso
(c) Falso
(d) Verdadeiro
2. *MUX*

SEÇÃO 11-17
1. (a) Verdadeiro
(b) Falso
2. Ele fornece endereços de linha para a DRAM durante os ciclos de refresh.
3. Multiplexação de endereços e operação de refresh
4. (a) Falso
(b) Verdadeiro

SEÇÃO 11-18
1. Não
2. Posições de memória com a mesma parte alta do endereço (mesma linha)
3. Apenas o endereço de coluna deve ser armazenado.
4. *Extended data output*
5. *Burst*
6. O clock do sistema

SEÇÃO 11-19
1. Dezesseis
2. Quatro
3. Falso; quando se expande a capacidade de memória, cada chip é selecionado por uma saída diferente do decodificador (vide Fig. 11-43).
4. Verdadeiro

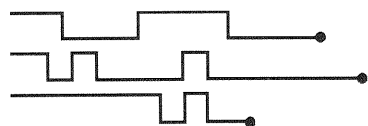
SEÇÃO 11-20
1. Alimentação a bateria para RAM CMOS; memória flash
2. Econômica
3. Os dados são lidos da memória na mesma ordem em que foram escritos.
4. Uma FIFO utilizada para transferir dados entre dispositivos com velocidades de operação muito diferentes

SEÇÃO 11-21
1. Evita glitches de decodificação desabilitando o decodificador enquanto as linhas de endereço estão mudando
2. Um modo de testar RAM escrevendo um padrão xadrez (primeiro 01010101 e depois 10101010) em cada posição de memória e lendo posteriormente. É usado porque detectará quaisquer curtos ou interações entre células adjacentes.
3. Um teste automático de RAM realizado pelo computador quando é energizado

SEÇÃO 11-22
1. Um código colocado na última ou duas últimas posições da ROM que representa a soma prevista para os dados de todas as outras posições da ROM. É utilizado como um modo de testar a ocorrência de erros em uma ou mais posições da ROM.

CAPÍTULO 12

Aplicações de um Dispositivo de Lógica Programável



■ SUMÁRIO

Os diagramas da GAL 16V8A apresentados neste capítulo foram reproduzidos por cortesia da Lattice Semiconductor Corporation, Hillsboro, Oregon.

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Identificar as aplicações dos dispositivos de lógica programável.
- Compreender a organização interna de um dispositivo de lógica programável típico.
- Utilizar ferramentas de software capazes de programar uma grande variedade de dispositivos de lógica programável.
- Implementar um projeto lógico completo usando PLDs.

■ INTRODUÇÃO

Ao longo deste livro, examinamos vários blocos básicos de sistemas digitais. À medida que cada bloco era descrito, um circuito integrado disponível era citado como um meio de se implementar a função lógica em questão. Muitos sistemas em operação hoje são combinações de circuitos integrados que realizam as funções para as quais o sistema foi especificado. Imagine por um momento que você é um projetista de circuitos e que desenvolveu um circuito que é composto de portas lógicas, decodificadores, contadores, registradores etc. Como você irá implementar este circuito? Como você vai demonstrar que o seu projeto funciona? Uma vez que os componentes tenham sido especificados e o diagrama esquemático final esteja desenhado, você provavelmente irá construir um protótipo.

Esse processo é, por natureza, sujeito a erros na conexão dos componentes. O uso de vários tipos diferentes de circuitos integrados tende a criar um grande número de conexões. Mesmo com técnicas de protótipo altamente organizadas, em geral é muito difícil identificar visualmente uma conexão no protótipo. Utilizando as técnicas de pesquisa de falhas apresentadas neste livro, você pode isolar o problema, mas corrigir uma ligação errada no protótipo pode ser bastante difícil.

Imagine que você fosse capaz de agrupar vários componentes em um único circuito integrado que foi "programado" para funcionar conforme especificado no seu projeto. Os dispositivos de lógica programável, que foram introduzidos no Cap. 11, oferecem essa funcionalidade. Com esses dispositivos, a possibilidade de erros se desloca das conexões para a programação do dispositivo. Conseqüentemente, um dispositivo que é capaz de ser apagado e reprogramado é ideal para situações em que mudanças e revisões serão feitas com freqüência. O objetivo deste capítulo é fornecer informações mais detalhadas sobre dispositivos de lógica programável (PLDs) que são reprogramáveis. Também discutiremos ferramentas e métodos para desenvolver projetos lógicos em circuitos funcionais usando lógica programável.

A maior parte deste capítulo é sobre um dispositivo particular, a GAL 16V8A. Ela foi selecionada por ser compatível com quase todos os dispositivos PAL, ter um custo razoável e ser apagável e reprogramável. Estes recursos são bastante úteis para ambientes de laboratório e no desenvolvimento de protótipos. Os conceitos que se aplicam a esse dispositivo são igualmente aplicáveis a outros tipos de PLDs. O CUPL, um dos vários pacotes de software usados para programar PLDs, também será descrito para permitir que você monte o seu sistema de desenvolvimento. Ele foi escolhido porque é fácil de usar e é bastante popular na indústria.

12-1 A GAL 16V8A (ARRANJO LÓGICO GENÉRICO)

A GAL 16V8A foi introduzida pela *Lattice Semiconductor*. Sua arquitetura é bem similar à dos dispositivos PAL descritos no Cap. 11 (Fig. 11-22). Conseqüentemente, ela pode ser usada como um substituto genérico, compatível pino a pino, para a maioria dos dispositivos PAL. Em vez de utilizar ligações que são programáveis uma única vez para selecionar as entradas que irão produzir os termos produto apropriados, os dispositivos GAL usam uma matriz EEPROM. Certas posições desta memória são designadas para controlar as conexões programáveis para a **matriz de entrada**. Cada bit na matriz de entrada representa uma conexão programável entre uma linha e uma coluna. Isto permite que o dispositivo possa ser apagado e reprogramado. A tecnologia CMOS permite que o componente opere com baixo consumo e velocidade comparável à dos dispositivos TTL. Felizmente não é necessário saber os endereços de cada bit na matriz. O software de programação se encarrega desses detalhes de uma maneira bastante amigável para o usuário.

O diagrama lógico completo da GAL 16V8A pode ser visto na Fig. 12-1. Este dispositivo tem oito pinos dedicados para entrada (pinos 2 a 9), dois pinos de entrada com funções especiais (pinos 1 e 11), e oito pinos que podem ser usados como entradas ou saídas (pinos 12 a 19). Os componentes principais de dispositivos GAL são a matriz de entrada, as portas AND, que geram os produtos das entradas; e as **macrocélulas da lógica de saída (OLMCs —*output logic macro cells*)**. Note que cada um dos oito pinos de entrada (2 a 9) estão conectados a uma coluna da matriz de entrada. O complemento de cada uma dessas entradas também está conectado a uma coluna da matriz de entrada. Esses pinos devem ser sempre identificados como entrada quando estivermos programando a 16V8A. Um nível lógico e seu complemento são realimentados de cada OLMC para uma coluna da matriz de entrada. A fonte desses níveis lógicos é determinada pela configuração de cada OLMC, como será visto mais adiante. Isto contabiliza 32 variáveis de entrada (colunas na matriz de entrada) que podem ser conectadas às 64 portas AND de múltiplas entradas.

No Cap. 11 vimos que a única linha mostrada como uma entrada das portas AND na verdade representa até 32 entradas em cada porta (vide Fig. 11-20). Qualquer coluna mostrada como estando conectada (designada por um X) a uma linha é uma entrada de uma porta AND, juntamente com qualquer outra coluna que esteja conectada a esta mesma linha. Na matriz de entrada, qualquer coluna pode

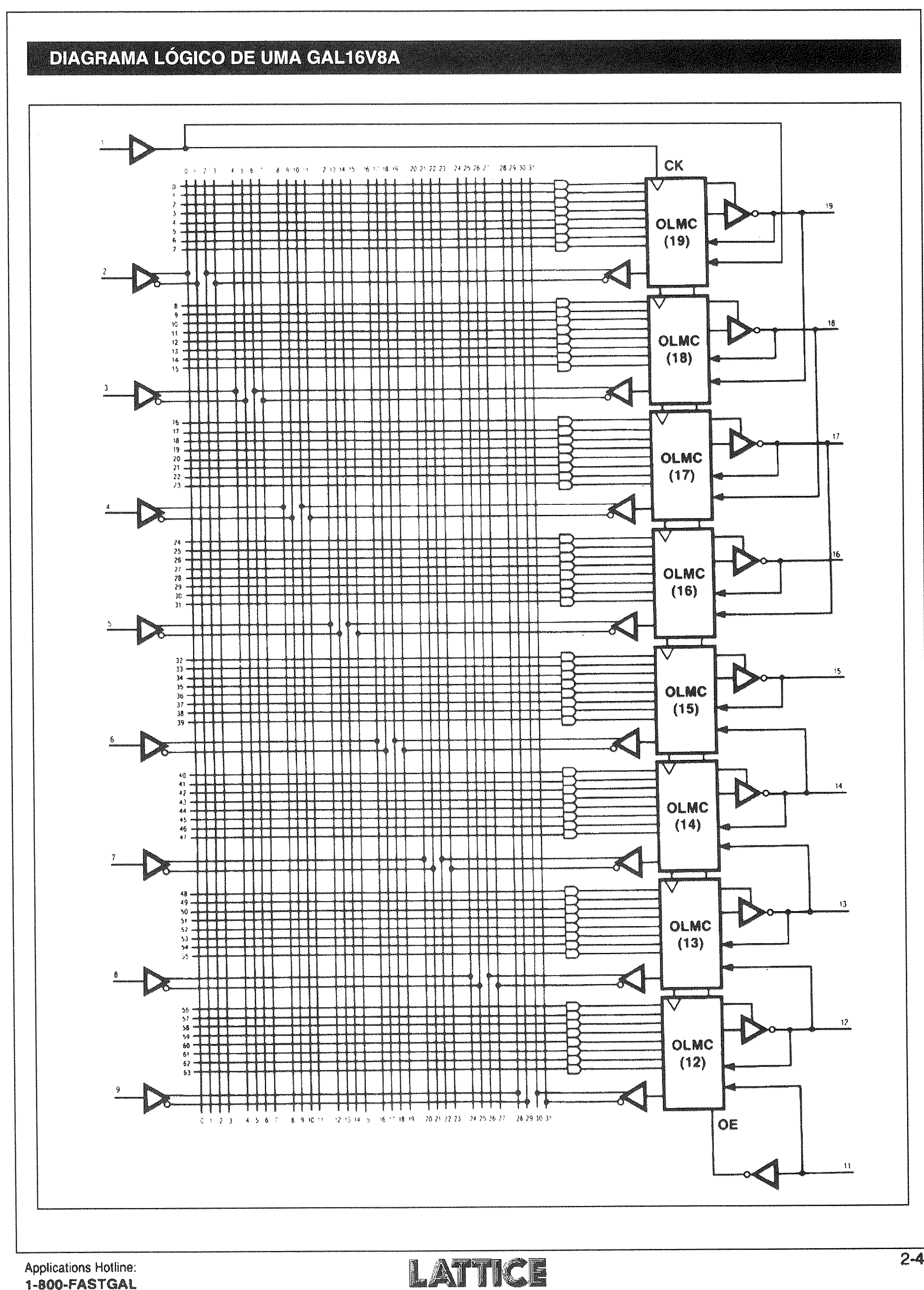


Fig. 12-1 Diagrama lógico de uma GAL 16V8A.

ser conectada à entrada de uma porta AND durante o processo de programação.

A flexibilidade da GAL 16V8A se baseia no fato de sua OLMC ser programável. Oito diferentes produtos (saídas das portas AND) são fornecidos como entradas de cada uma das macrocélulas. Dentro de cada OLMC, é gerada uma expressão na forma de soma de produtos, fazendo uma operação OR com esses produtos. Lembre-se de que no Cap. 4 vimos que qualquer função pode ser escrita na forma de soma de produtos. Dentro da OLMC, o resultado dessa soma de produtos pode ser direcionado para um pino de saída para implementar um circuito combinacional ou pode ser sincronizada com o auxílio de um flip-flop D para implementar um circuito seqüencial.

Para compreender a operação de uma OLMC, veja a Fig. 12-2, que mostra a estrutura da OLMC(n), onde n representa um número entre 12 e 19. Observe que sete dos produtos estão incondicionalmente conectados às entradas da porta OR. O oitavo termo produto é conectado a um multiplexador de termos produto de duas entradas (PTMUX) que aciona a oitava entrada da porta OR. O oitavo termo produto também é conectado a uma das quatro entradas de um multiplexador de quatro entradas (TSMUX) que habilita o inversor com saída tristate, que, por sua vez, aciona o pino de saída [I/O(n)]. O multiplexador de saída (OMUX) é um MUX de duas entradas que seleciona entre a saída combinacional (porta OR) e a saída do flip-flop D. Um quarto multiplexador seleciona se o sinal lógico deve ser realimentado para a matriz de entrada. Este é chamado de multiplexador de *realimentação* (FMUX).

Cada um desses multiplexadores é controlado por bits programáveis na EEPROM. É desse modo que a configuração da OLMC pode ser alterada pelo programador. Um outro bit que é programável é o da entrada da porta XOR. Ele possibilita que a polaridade da saída seja programável, conforme foi visto no Cap. 11 (veja Fig. 11-23). Lembre-se de que a porta EX-OR pode ser usada para complementar um sinal lógico de modo seletivo, como pode ser visto na Fig. 12-3.

Podemos compreender as várias opções de configuração estudando as possíveis entradas de cada um dos multiplexadores. O TSMUX controla a entrada de habilitação do buffer tristate. Se a entrada V_{CC} está selecionada, a saída está sempre habilitada, como uma porta lógica combinacional comum. Se a entrada aterrada está selecionada, a saída tristate do inversor está sempre em alta impedância (permitindo que seu pino I/O seja utilizado como entrada). Uma outra entrada que pode ser selecionada vem da entrada *OE* que está no pino 11. Isto permite que a saída seja habilitada ou desabilitada por um sinal lógico externo aplicado ao pino 11. A última entrada que pode ser selecionada é o termo produto proveniente da oitava porta AND. Isto permite que uma combinação AND de termos da matriz de entrada possa habilitar ou desabilitar a saída.

O FMUX seleciona o sinal que é realimentado para a matriz de entrada. Neste caso, existem três seleções possíveis. Selecionando a entrada do *MUX* que está conectada ao estágio adjacente ou a entrada do *MUX* conectada ao seu próprio pino de I/O, podemos realimentar um estado existente para a matriz de entrada em alguns modos de operação. Esse recurso fornece à GAL 16V8A a habilidade de implementar circuitos seqüenciais como o circuito latch com portas NAND descrito no Cap. 5. Essas opções de realimentação também permitem que pinos de I/O sejam usados como entradas dedicadas em vez de saídas. Um desses dois caminhos de realimentação é escolhido, dependendo do modo para o qual o chip foi programado. A terceira opção

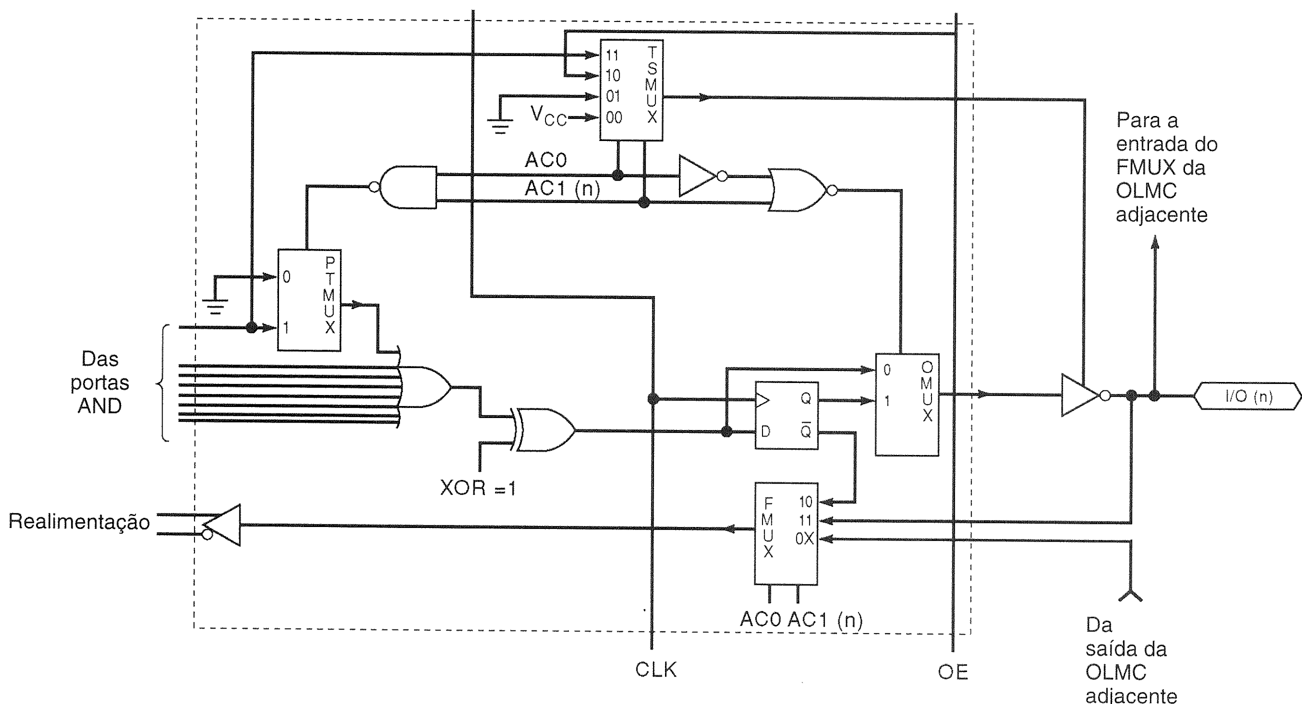
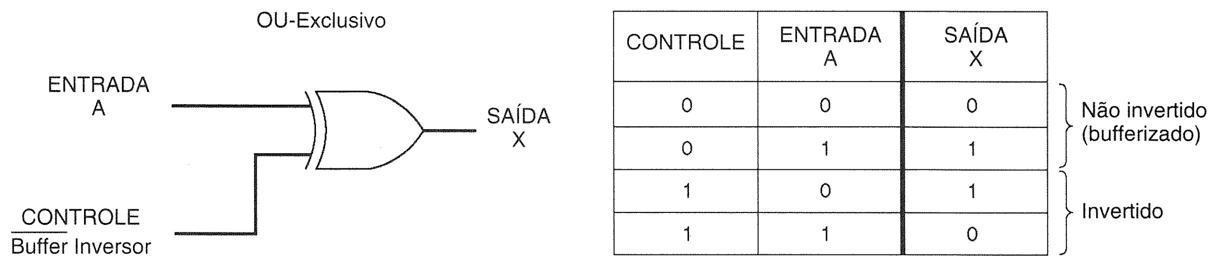


Fig. 12-2 Macrocélula da lógica de saída de uma GAL 16V8A.



CONTROLE	ENTRADA A	SAÍDA X
0	0	0
0	1	1
1	0	1
1	1	0

Fig. 12-3 Usando um EX-OR para complementar de modo seletivo.

seleciona a saída de um flip-flop D, permitindo que o estado atual do flip-flop (que pode ser usado para determinar o próximo estado) seja realimentado para a matriz de entrada. Isso permite que circuitos seqüenciais sejam implementados, tais como contadores e registradores de deslocamento.

Com todas essas opções, parece que existe uma longa lista de configurações possíveis. Na prática, todas as decisões de configuração serão feitas pelo software, como veremos mais adiante neste capítulo. Na verdade, a GAL 16V8A tem apenas três modos, mas é muito importante compreender como o hardware é configurado em cada um deles para que seja possível entender as capacidades e as limitações do dispositivo. Os três modos de configuração são: (1) *modo simples,* que é usado para implementar expressões na forma de soma de produtos sem saídas do tipo tristate; (2) *modo complexo,* que implementa expressões na forma de soma de produtos com saídas do tipo tristate que são habilitadas por um produto AND, e (3) *modo registrador,* que permite que OLMCs individuais operem em uma configuração combinacional com saídas tristate (semelhante ao modo complexo) ou em modo síncrono com flip-flops D sincronizados por um sinal de clock comum.

A linha 60 da matriz EEPROM armazena a *palavra de controle da arquitetura.* Cada bit desta palavra tem um papel na determinação da configuração de hardware das OLMCs. A Fig. 12-4 mostra como esses bits estão organizados. A não ser que você vá programar manualmente a GAL 16V8A (o que quase nunca é feito), não terá que se preocupar com a organização dos bits nessa palavra de controle. O software de programação se encarregará disso. Você deve, entretanto, compreender as funções de cada bit. Os bits SYN e AC0 são usados para especificar um dos três modos de configuração para todo o chip. Os bits AC1(*n*) (um por OLMC) determinam a operação de cada OLMC individual naquele módulo. Os oito bits XOR(*n*) são usados para complementar seletivamente as respectivas saídas das OLMCs. Os 64 bits de desabilitação de termos produto (PTD) são usados para desabilitar as portas AND que não estão sendo usadas.

Modo Simples

O chip está programado no modo simples quando SYN = 1 e AC0 = 0 na palavra de controle. Neste modo, duas configurações são possíveis para cada macrocélula (OLMC):

Entrada dedicada [quando AC1(*n*) = 1]
Saída combinacional dedicada [quando AC1(*n*) = 0]

Cada OLMC pode ser programada de modo independente em uma dessas duas configurações. A configuração de entrada dedicada conecta um pino de I/O de uma OLMC na matriz de entrada através do redirecionamento do sinal por uma OLMC adjacente, como por exemplo o pino 18 na Fig. 12-5. O inversor tristate deste pino está incondicionalmente desabilitado (alta impedância). Os pinos 11 e 1 servem como entradas que são redirecionadas através das OLMCs 12 e 19, respectivamente. Neste modo, os pinos 15 e 16 não podem ser usados como entradas dedicadas, uma vez que eles não têm uma conexão com uma célula adjacente.

A configuração com saída dedicada para uma OLMC tem seu inversor tristate sempre habilitado, como pode ser visto na Fig. 12-5 para a OLMC 19. Esta é a única configuração capaz de agrupar os oito termos produtos em uma expressão na forma de soma de produtos para gerar uma saída combinacional. As OLMCs 15 e 16 estão sempre na configuração de saída dedicada sem realimentação no modo simples. As outras seis OLMCs podem ter suas saídas realimentadas para a matriz de entrada por meio de uma célula adjacente. Por exemplo, observe que a saída da OLMC 19 é realimentada para a matriz de entrada através da OLMC 18.

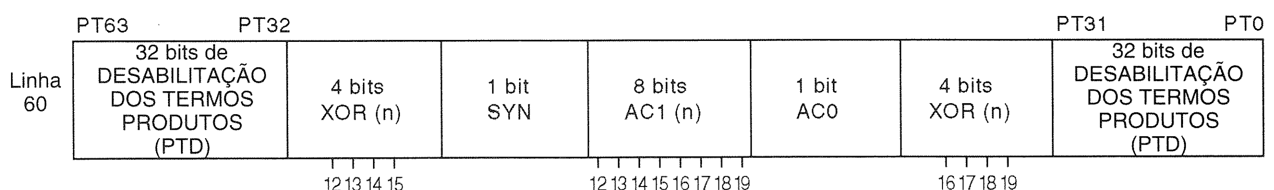


Fig. 12-4 A arquitetura da palavra de controle da GAL 16V8A.

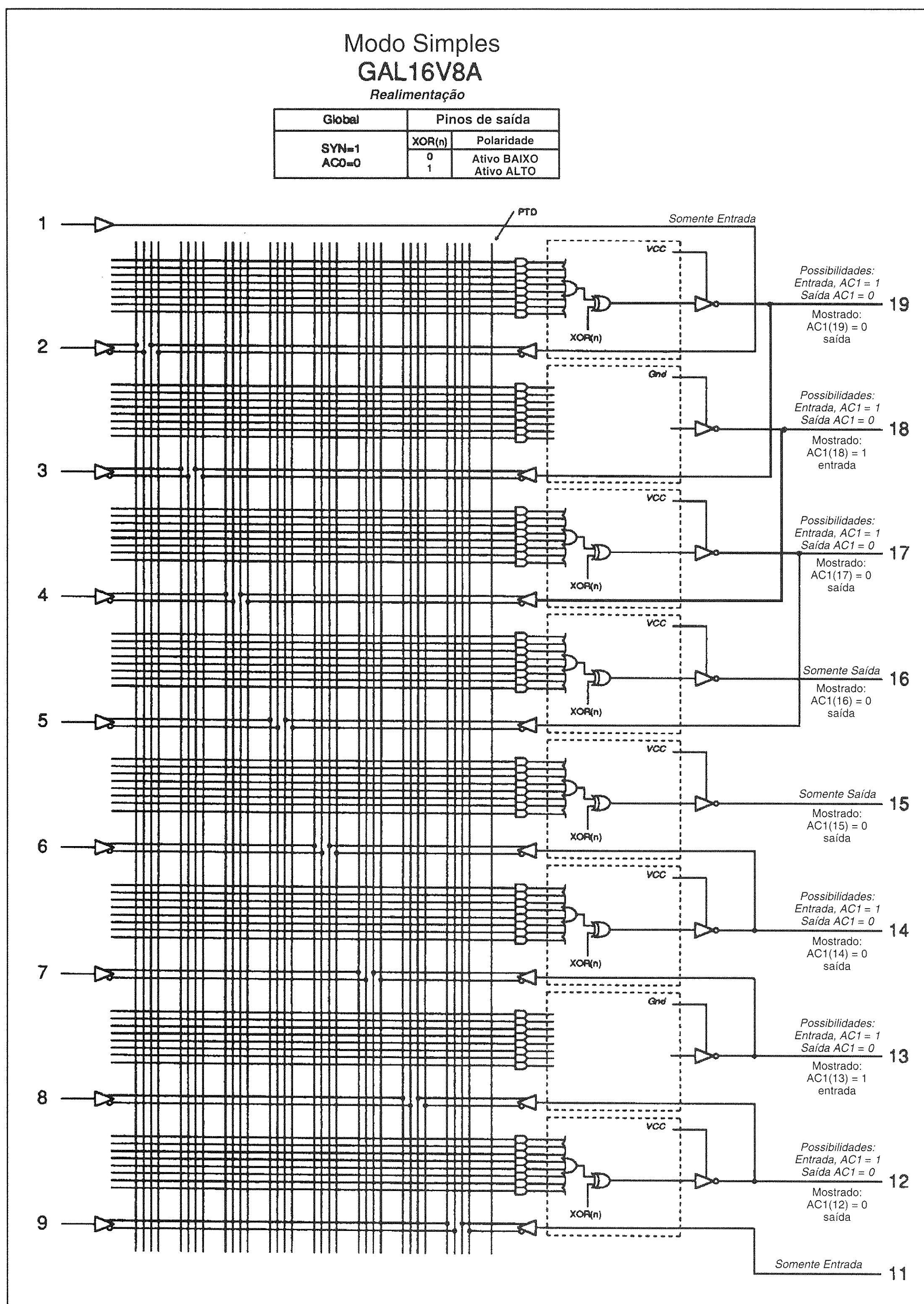


Fig. 12-5 Configurações das OLMCs no modo simples.

EXEMPLO 12-1

O circuito do Cap. 3, Fig. 3-17, pode ser representado por uma expressão booleana $y = AC + B\overline{C} + \overline{A}BC$. Se a GAL 16V8A fosse usada para implementar este circuito, que modo seria usado, e como o hardware seria programado?

Solução

Esta expressão é uma soma de produtos simples sem necessidade de saídas do tipo tristate, portanto o modo simples é usado (SYN = 1, AC0 = 0). O hardware está configurado como mostrado na Fig. 12-6. A OLMC 19 está configurada como uma saída dedicada [AC1(19) = 0]. Observe as conexões da matriz que gera a expressão desejada. As linhas de entrada de todas as portas AND não-utilizadas são na verdade conectadas a todas as colunas. Isso assegura que a saída da porta AND seja igual a 0, uma vez que ela tem como entradas as variáveis e seus complementos. Os símbolos de conexão (X) foram omitidos nesse diagrama para maior clareza.

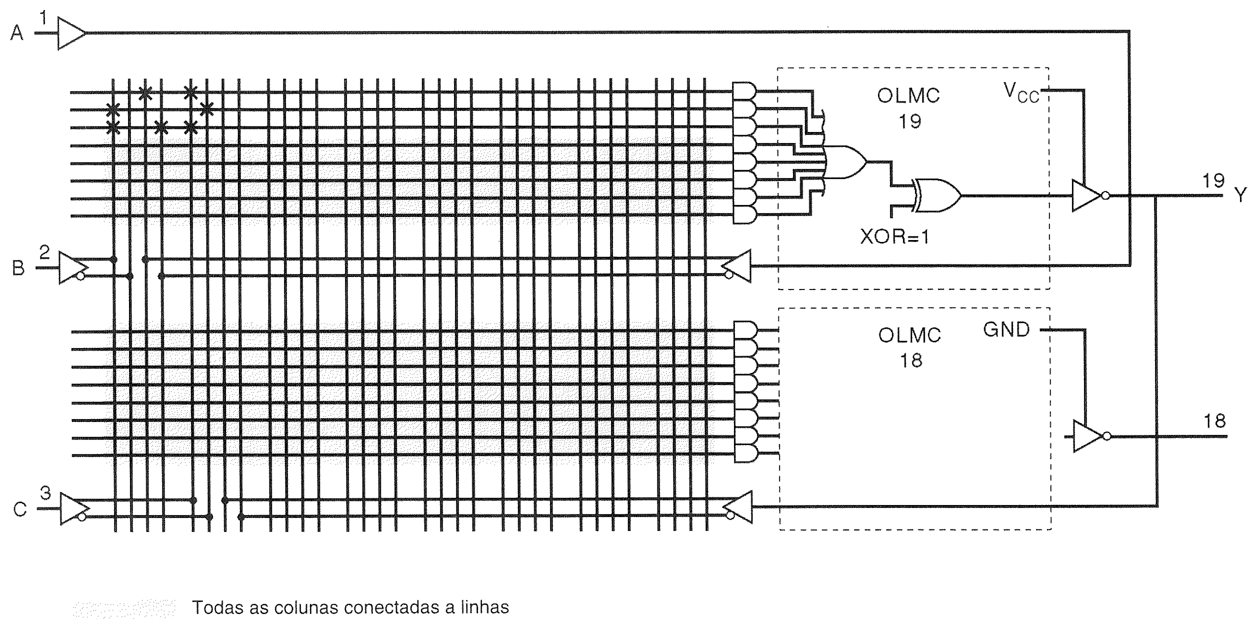


Fig. 12-6 Implementação de lógica combinacional simples.

Por que é tão importante saber tantos detalhes se o software vai se encarregar de toda a programação? O projetista deve decidir quais os pinos que serão conectados (e programados) como entradas e como saídas. Se o projetista declarar o pino 15 como uma entrada e também especificar oito termos produtos diferentes em uma única expressão, o software vai gerar uma mensagem de erro. Isso acontece porque o modo simples é necessário para se ter oito termos produtos, mas o pino 15 não pode ser usado como uma entrada neste modo. Entretanto, se o pino 15 for usado como saída e o pino 14 como entrada, o software pode programar o dispositivo para operar no modo simples.

Modo Complexo

O modo complexo é definido por SYN = 1 e AC0 = 1. O bit AC1 para cada OLMC está no nível lógico 1. A Fig. 12-7 mostra o diagrama lógico para a 16V8A. O inversor tristate que aciona o pino de saída é habilitado por uma expressão produto. Em outras palavras, uma expressão lógica deve ser especificada para habilitar a saída. Isto deixa sete termos produtos, que podem ser usados para formar uma expressão como soma de produtos. Os estados lógicos dos pinos de saída das OLMCs 13 a 18 também são realimentados através de sua respectiva OLMC para a matriz de entrada. As OLMCs 12 e 19 não podem ser realimentadas e nem podem ser usadas como entradas, uma vez que o caminho de realimentação é usado para permitir que os pinos 11 e 1 possam ser usados como entradas. Nesta figura, o pino 19 é mostrado como uma saída dedicada sem realimentação, o pino 18 é uma entrada, e o pino 17 é uma saída com realimentação. Este pino também pode servir como entrada quando o inversor tristate estiver desabilitado.

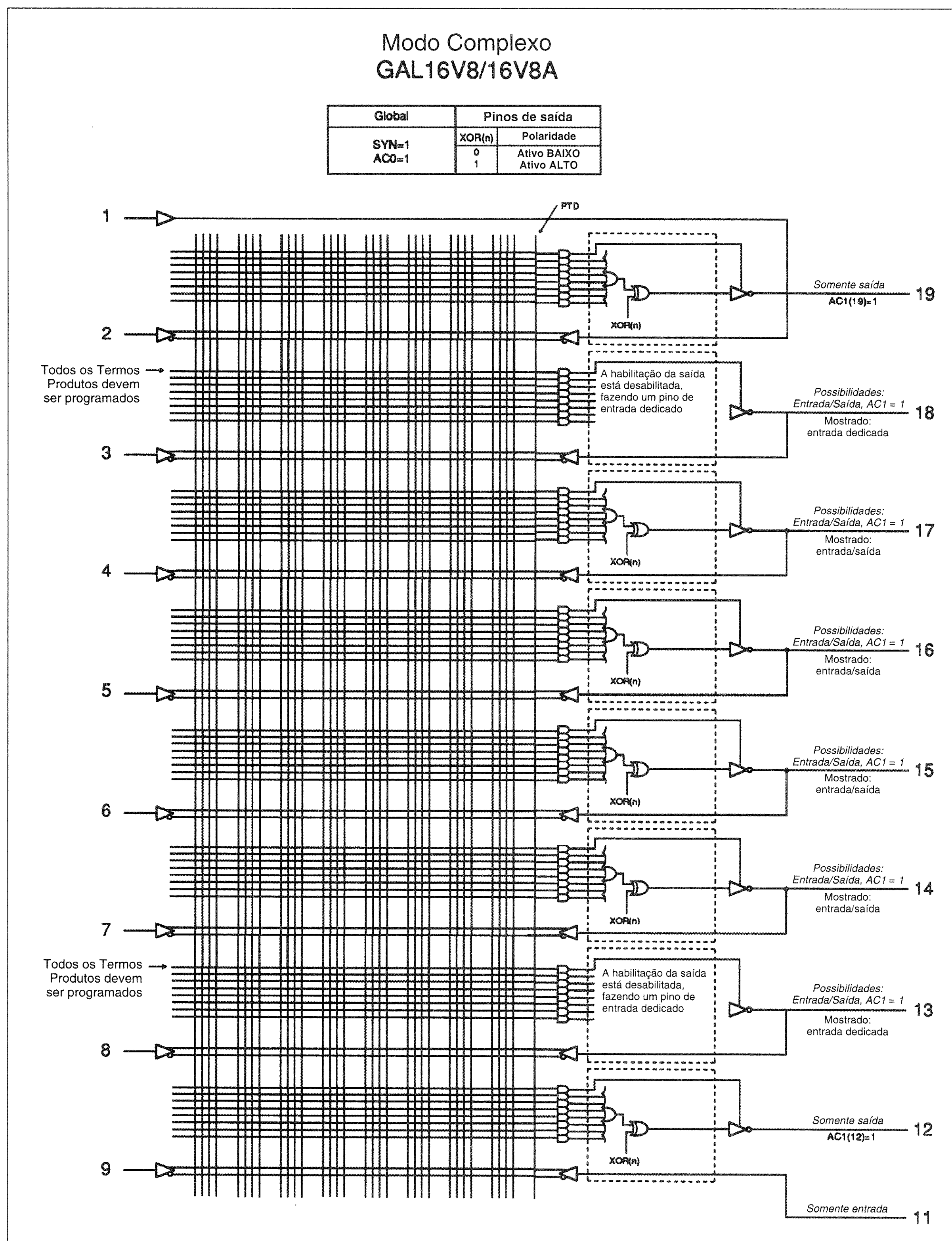


Fig. 12-7 Configurações das OLMCs no modo complexo.

EXEMPLO 12-2

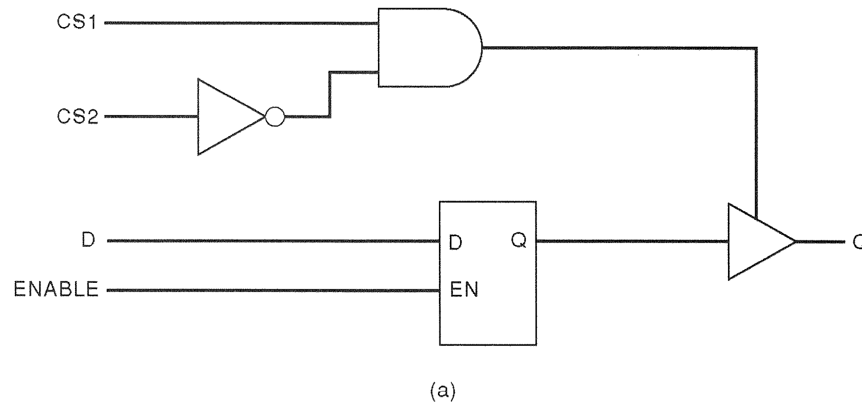
Determine o modo de operação, escreva as equações lógicas e desenhe a configuração de hardware de um latch transparente tipo *D* com saídas tristate habilitadas por *CS*1 e *CS*2, como se pode ver na Fig. 12-8(a).

Solução

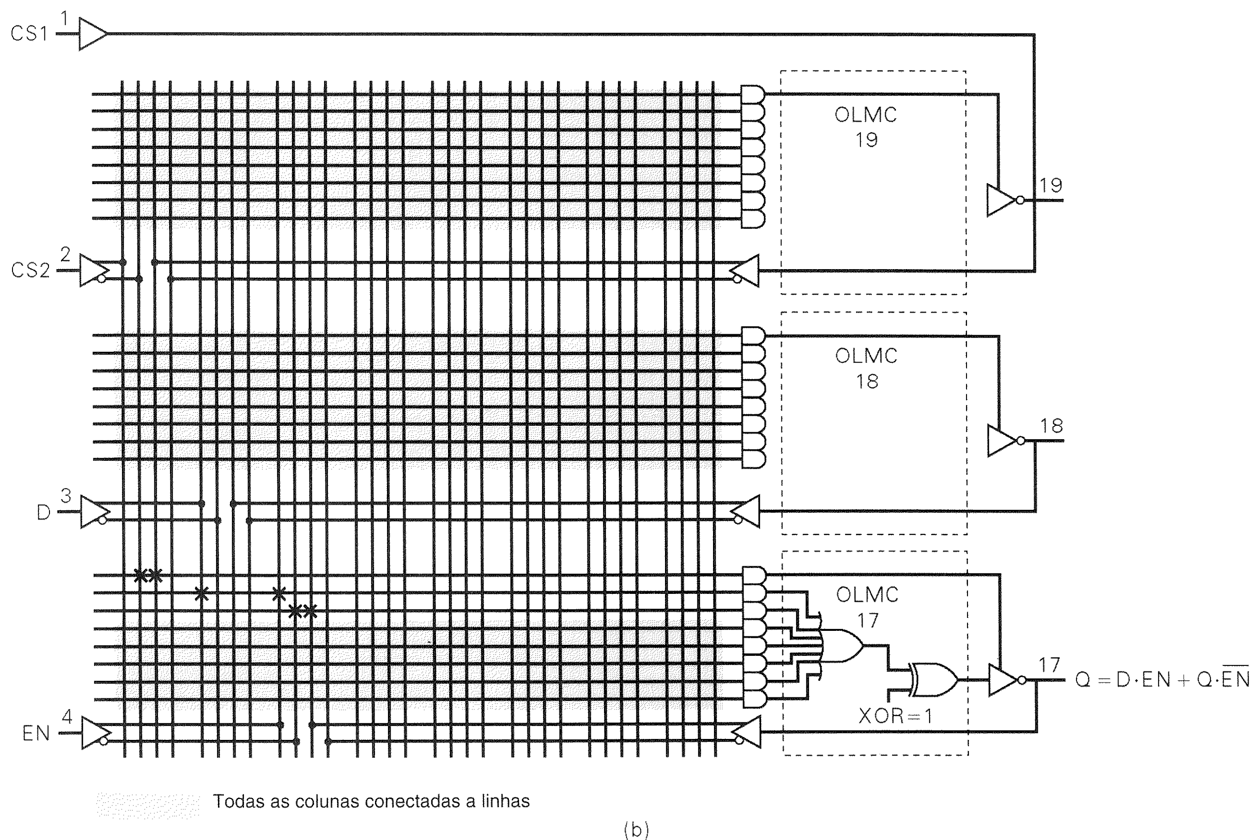
Uma vez que saídas síncronas não são necessárias, mas saídas tristate estão especificadas, o modo complexo é usado. As equações são:

$$\text{habilitação da saída} = CS1 \cdot \overline{CS2}$$
$$Q = D \cdot EN + Q \cdot \overline{EN}$$

A configuração da GAL 16V8A é mostrada na Fig. 12-8(b) com a OLMC 17 usada para gerar sinal *Q*. Observe as conexões na matriz para implementar as equações.



(a)



(b)

Fig. 12-8 (a) Latch transparente com saídas tristate. (b) Implementação do latch no modo complexo.

Modo Registrador

O modo registrador é caracterizado por SYN = 0 e AC0 = 1. Existem duas configurações possíveis para cada OLMC neste modo:

1. *Configuração síncrona* (quando AC1 = 0)
2. *Configuração combinacional* (quando AC1 = 1)

A operação da configuração combinacional (Fig. 12-9, OLMCs 12 e 18) é semelhante à operação de uma OLMC no modo complexo. A habilitação da saída é controlada por

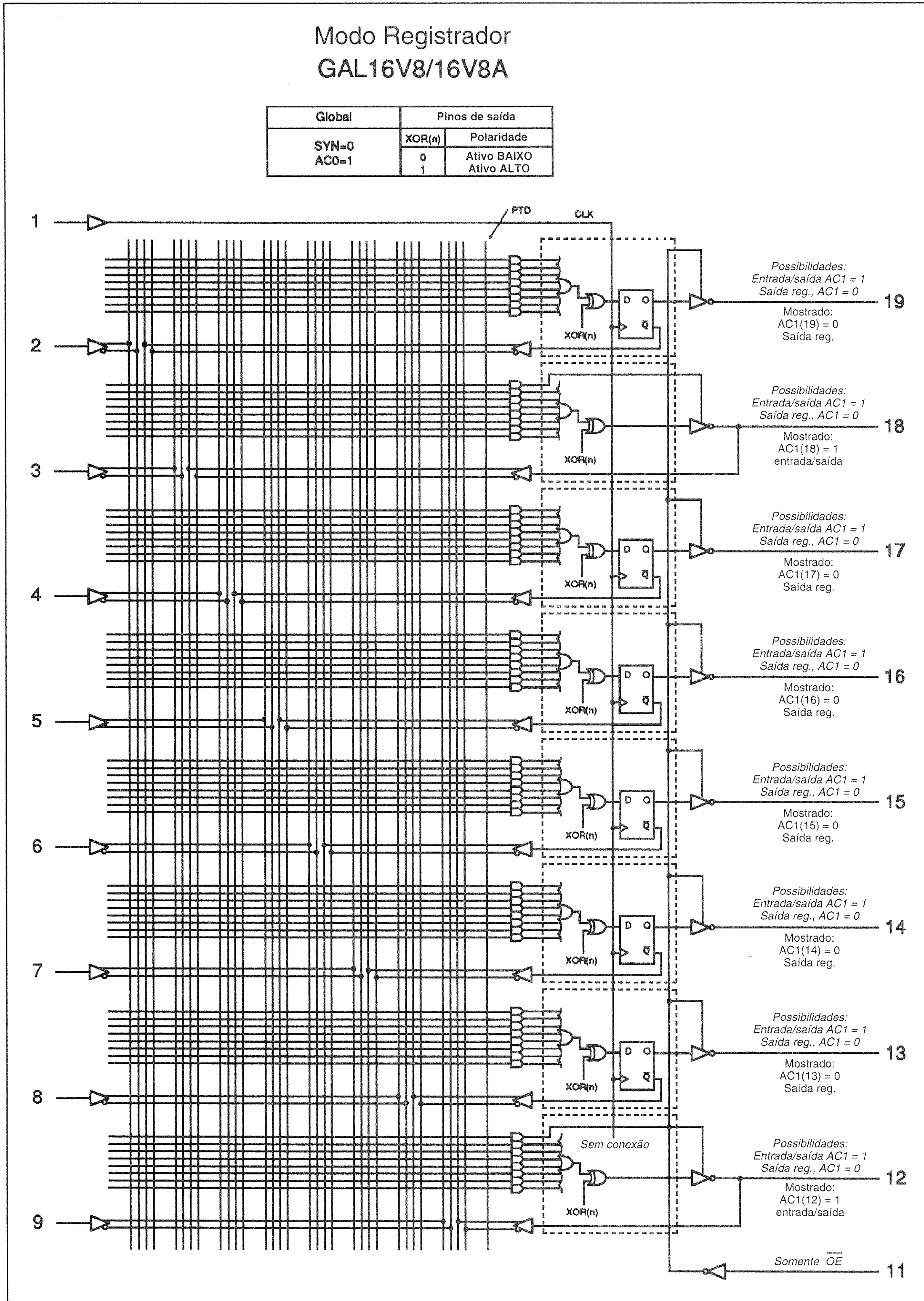


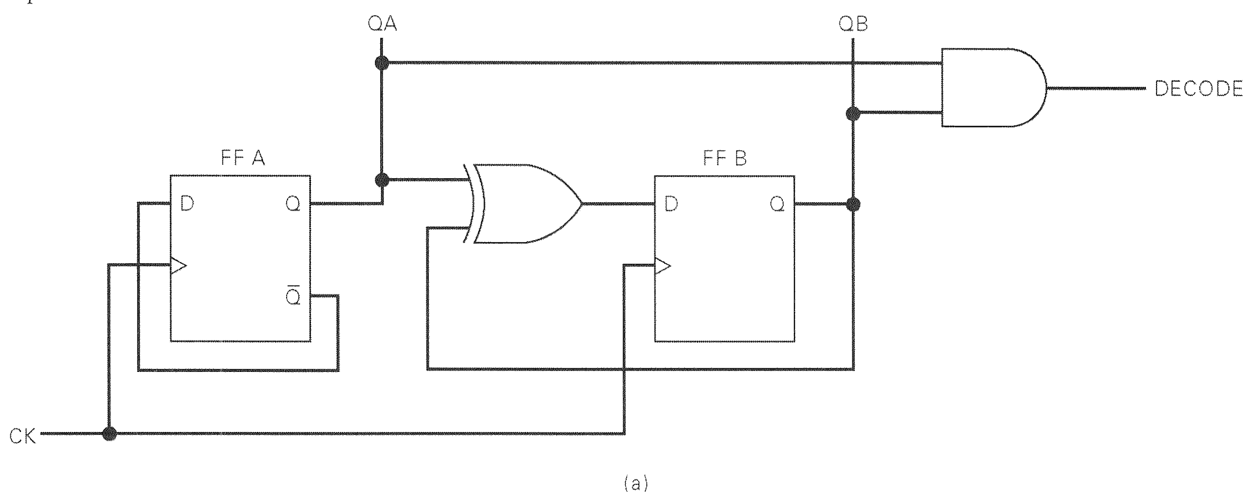
Fig. 12-9 Configurações das OLMC no modo registrador.

uma expressão produto da matriz de entrada. Isto fornece habilidade de usar sinais bidirecionais em aplicações de barramentos. O sinal de saída também é realimentado através de sua OLMC correspondente.

A configuração síncrona (OLMCs 13 a 17 e 19) usa um flip-flop D em cada OLMC para sincronizar todas as saídas dos flip-flops à transição de um sinal de clock comum. O pino 1 é uma entrada dedicada para o sinal de clock. Todos os inversores tristate para as OLMCs que foram configuradas com saídas seqüenciais são controlados pelo pino 11 $(\overline{OE})$ Estas saídas seqüenciais podem ter até oito termos produtos em uma expressão lógica que aciona a entrada *D* do flip-flop.

EXEMPLO 12-3

Determine o modo de operação de uma GA 16V8A, escreva as equações lógicas e desenhe a configuração de hardware para um contador de módulo 4 de 2 bits com uma saída decodificada para o estado *QB, QA* = 11, conforme a Fig. 12-10(a).



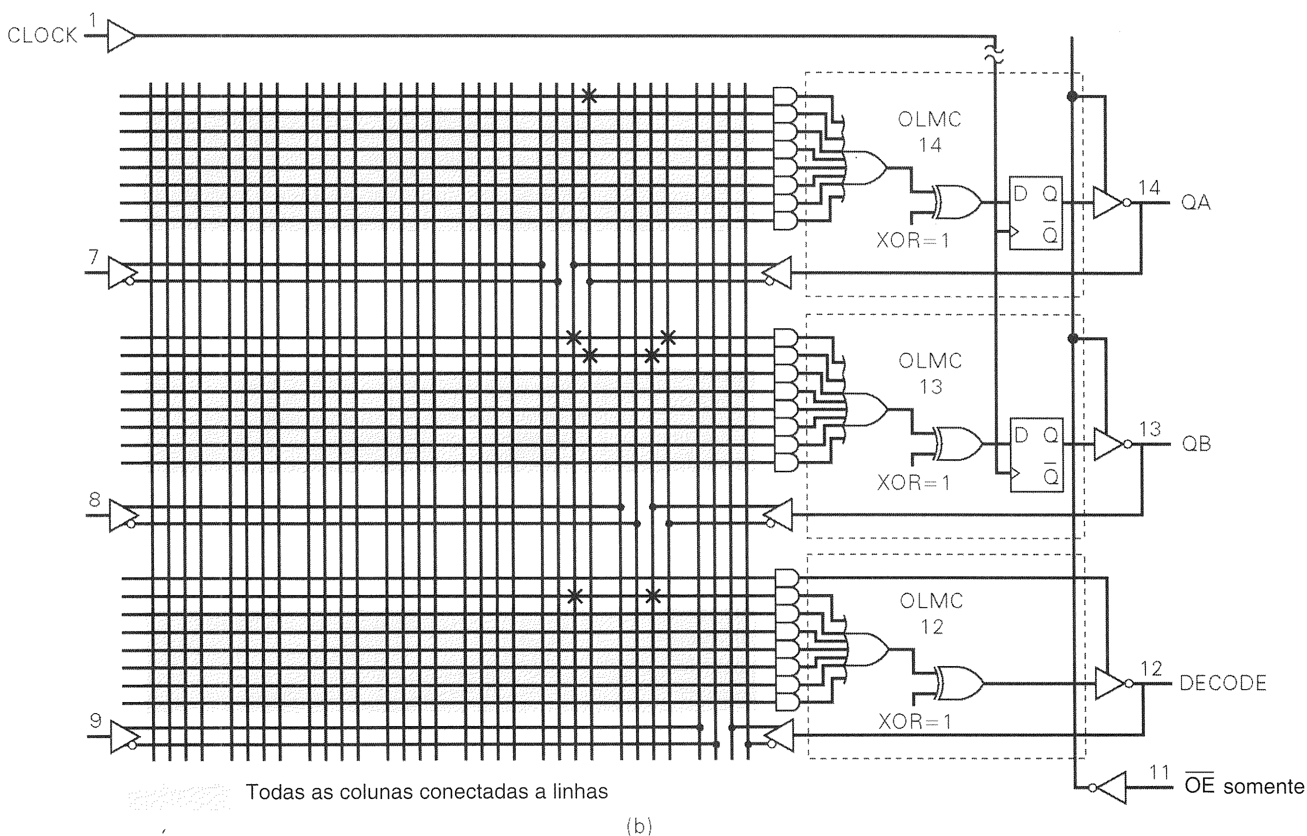


Fig. 12-10 (a) Contador binário de módulo 4 com saída decodificada; (b) implementação do contador no modo registrador.

Solução

Um contador de módulo 4 necessita do modo registrador. As equações para as entradas dos flip-flops *D* e para a saída decodificada são:

$D_A = \overline{QA}$ (a entrada *D* do FF *A*)
$D_B = \overline{QB} \cdot QA + QB \cdot \overline{QA} = QB \oplus QA$ (a entrada *D* do FF *B*)
$\text{DECODE} = QB \cdot QA$

A configuração de hardware pode ser vista na Fig. 12-10(b). Você deve verificar que as expressões foram implementadas corretamente.

Questões de Revisão

1. Diga duas vantagens dos dispositivos GAL em relação aos dispositivos PAL.
2. Diga os três modos de operação de uma GAL 16V8A.
3. Dê o nome de cada configuração da OLMC em cada modo.
4. Em qual modo a saída tristate é controlada pelo pino 11?
5. Em qual modo o pino 1 é dedicado para a entrada de clock?

12-2 PROGRAMANDO PLDs

Estudamos a programação de PROMs, EPROMs, EEPROMs e PLDs no Cap. 11. Obviamente, deve existir uma maneira melhor de especificar a programação de um PLD do que desenhar **X**'s e **O**'s no diagrama da matriz. Para compreender e apreciar o papel do software de desenvolvimento, vamos examinar alguns detalhes do processo de programação de um PLD.

Vários elementos são necessários para projetar e construir circuitos usando PLDs:

- Um computador (PC)
- Software de desenvolvimento para PLDs
- Programador
- Software para acionar o programador
- Dispositivo de lógica programável

A Fig. 12-11 mostra um sistema de desenvolvimento típico. O projeto inicial é fornecido como entrada ao software de desenvolvimento, conforme será visto mais adiante. Este software traduz o projeto inicial em um arquivo chamado "mapa de fusíveis". Este mapa indica quais fusíveis devem permanecer abertos e quais devem ficar intactos. O mapa de fusíveis é então transformado para um outro arquivo de saída, cujo formato é mais adequado para a sua transmissão para o programador.

O software de programação que se comunica com o programador é então chamado no PC. Isso permite que o usuário informe ao programador o tipo de dispositivo que vai ser programado. O arquivo de saída é enviado através de um cabo para o programador. Este processo é chamado de **downloading**. Finalmente, o PLD é colocado no soquete do programador e um comando é enviado do computador para programar o PLD. Os programadores geralmente têm um soquete especial que permite que você coloque o chip e depois prenda os pinos entre seus contatos. Ele é chamado de **soquete ZIF** (**ZIF** — ***zero-insertion force***). Programadores universais que podem programar praticamente todos os tipos de dispositivos, incluindo PROMs, EPROMs, microcontroladores e vários tipos de PLDs, estão disponíveis por diversos fabricantes.

Felizmente, à medida que os componentes programáveis começaram a proliferar, os fabricantes viram a necessidade de padronizar as pinagens e os métodos de programação. Um dos resultados dessa necessidade foi a criação do Conselho Unificado de Engenharia de Dispositivos Eletrônicos (**JEDEC** — Joint Electronic Device Engineering Council). Uma de suas realizações foi o padrão JEDEC 3, um formato padrão para transferência de dados de programação para PLDs, independentemente do fabricante ou do software de programação. As pinagens para os vários encapsulamentos de circuitos integrados também foram padronizadas, tornando os programadores universais menos complicados. Conseqüentemente, os programadores são capazes de programar vários tipos de PLDs. O software que permite ao projetista especificar a configuração para um PLD precisa simplesmente produzir um arquivo de saída segundo o padrão JEDEC. Então este arquivo JEDEC pode ser carregado em qualquer programador de PLDs compatível com JEDEC e que seja capaz de programar o tipo de PLD desejado.

Questões de Revisão

1. Diga os nomes de dois pacotes de software que são usados para implementar circuitos com PLDs.
2. Defina os seguintes termos:
   (a) *mapa de fusíveis* **(b)** *arquivo de saída*
3. O que é um arquivo JEDEC?

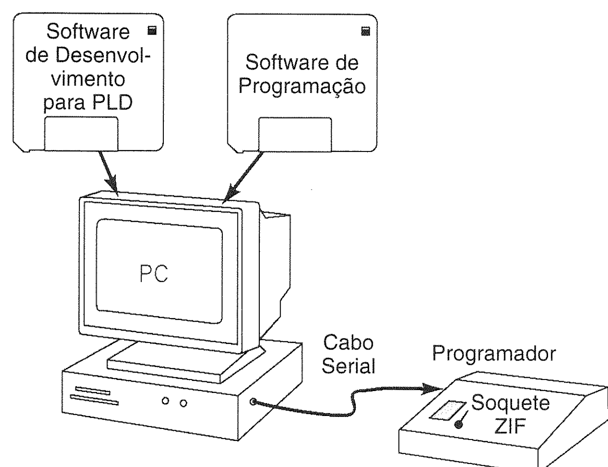


Fig. 12-11 Um sistema de desenvolvimento para PLDs.

12-3 SOFTWARE DE DESENVOLVIMENTO

O processo de gerar um arquivo JEDEC para depois transferi-lo para o programador pode ser um processo extremamente tedioso se for feito manualmente. Nestes últimos vinte anos, muitos pacotes de software foram desenvolvidos para permitir aos usuários um modo conveniente de projetar seus circuitos e gerar automaticamente o arquivo JEDEC. O primeiro software desse tipo, desenvolvido pela Monolithic Memories, Inc. (MMI), foi chamado PALASM, que é uma abreviação de PAL *assembler* (montador para PALs). Versões desse software ainda podem ser encontradas em domínio público para desenvolvimento de projetos simples com PLDs. Entretanto, o mercado tem produzido softwares para projeto com PLDs muito mais poderosos. Essas ferramentas são conhecidas como **compiladores** lógicos. Montadores e compiladores são programas que rodam em computadores pessoais. O usuário fornece um arquivo de entrada em um formato que o programa pode interpretar. A principal diferença entre montadores e compiladores está no método usado para relação entre as entradas e as saídas. Um montador necessita de um arquivo de entrada que define de um modo bastante específico a operação do dispositivo em termos que estão muito próximos ao hardware que está sendo programado, como por exemplo equações lógicas booleanas. Um compilador pode aceitar uma representação mais abstrata do mesmo projeto e traduzi-la em termos mais próximos ao hardware que está sendo programado.

Todos os montadores e compiladores permitem que o usuário forneça seu projeto na forma de equações booleanas que definem a saída em função das entradas disponíveis. Neste caso, o arquivo de entrada é um arquivo texto no formato ASCII, que pode ser gerado em qualquer editor genérico do seu computador. Neste arquivo estão especificados, juntamente com outras informações, o nome do projetista, a data e o número da versão do projeto. Os pinos a serem utilizados também devem estar especificados e denominados com os nomes dos sinais de entrada e de saída. A seguir, um conjunto de equações booleanas é relacionado e deve especificar o funcionamento lógico desse dispositivo. A maioria dessas ferramentas é capaz de simplificar as equações que estão no arquivo de entrada, utilizando algoritmos similares àqueles que você aprendeu nos capítulos anteriores. Este recurso auxilia o projetista, pois permite que a lógica a ser implementada seja especificada na sua forma mais simples, sem nenhuma tentativa de minimização das equações. Se a forma minimizada necessitar de um maior número de portas que aquele que o dispositivo programável possui, uma mensagem de erro é enviada.

Os compiladores de alto nível têm diversas opções para especificar o comportamento do circuito. Uma dessas opções é chamada de *captura do esquemático*. Este modo permite que a operação do circuito seja especificada simplesmente por meio do desenho de um diagrama esquemático de um circuito lógico usando um CAD. Este arquivo gráfico é traduzido em uma lista de ligações que especifica não só os tipos de componentes utilizados mas também como eles estão conectados. Vários programas diferentes podem ser necessários para colocar os vários tipos de desenhos em CADs em um formato que possa ser aceito pelo compilador. Esse método é bastante útil quando queremos implementar um circuito já existente em PLDs.

Como você deve ter aprendido nos capítulos anteriores, os circuitos lógicos são projetados a partir de tabelas-verdade e de tabelas de estado. Estas tabelas são utilizadas para simplificar o problema e produzir as expressões lógicas que são usadas para implementar o circuito real. Você também sabe que esse processo pode ser longo e tedioso. Os compiladores lógicos permitem que você entre com o seu projeto na forma de tabelas-verdade ou de tabelas de estado. Alguns compiladores podem até mesmo aceitar diagramas de tempo como entrada. O compilador passa a se encarregar de todos os passos de redução, da geração das expressões lógicas e da produção do tipo apropriado de arquivo de saída para programação do PLD que você tem em mente.

Esses métodos de entrada de dados para projetos com PLDs têm servido muito bem às necessidades da indústria durante o período de transição dos chips lógicos discretos para o uso de dispositivos programáveis. Durante a última década, os sistemas digitais tornaram-se cada vez mais complexos. No início dos anos 1980, o Departamento de Defesa dos Estados Unidos estava trabalhando no seu programa de circuitos integrados de velocidade muito alta (VHSIC — *Very High-Speed Integrated Circuit*). Durante esse programa, foi desenvolvida uma "linguagem de descrição de hardware" para especificar esses sistemas muito complexos. Essa linguagem evoluiu para um padrão do IEEE de descrição de circuitos lógicos para fins de projeto e documentação. Ela é conhecida como VHDL, Linguagem de Descrição de Hardware VHSIC (VHDL — VHSIC *Hardware Description Language*). Pode-se notar que ela é uma ferramenta de alto nível porque seu acrônimo já contém um outro acrônimo. A VHDL e outras linguagens de descrição de hardware parecem ser a direção para a qual a indústria está se movendo para a descrição de projetos com PLDs, especialmente para sistemas mais complexos que usam CPLDs e FPGAs.

Questões de Revisão

1. Descreva a diferença entre um montador e um compilador.
2. Diga quatro maneiras de se entrar com um projeto em um compilador.
3. O que a sigla *VHDL* significa?

12-4 COMPILADOR UNIVERSAL PARA LÓGICA PROGRAMÁVEL (CUPL)

Muitos programas para desenvolvimento com PLDs estão disponíveis. Todos eles compartilham algumas semelhanças, assim como possuem diferenças que os distinguem uns dos outros. Para fornecer exemplos reais de desenvolvimentos simples com PLDs, utilizaremos um dos compiladores lógicos mais populares disponíveis atualmente. Ele é chamado CUPL, e é fornecido pela Logical Devices. O CUPL é capaz de produzir arquivos para programação de uma grande variedade de dispositivos. Ele também possui muitos dos modos de entrada de dados descritos anteriormente.

O arquivo de entrada contém um cabeçalho padrão que é usado pelo software para produzir o arquivo de saída e para fornecer a documentação do projeto. A Fig. 12-12 mostra um modelo para um arquivo de entrada típico para o compilador CUPL, incluindo o cabeçalho. Normalmente, um modelo de arquivo de entrada como este é criado e salvo utilizando um editor de texto. Cada vez que um novo projeto é iniciado, o modelo é editado para que as lacunas sejam preenchidas. Observe que o /* e o */ são usados para delimitar os comentários; estes, por sua vez, vão facilitar a compreensão do arquivo de entrada. Observe também que toda linha de comando é delimitada por ponto-e-vírgula (;).

O CUPL permite que a declaração de cada pino seja fornecida de modo individual ou por meio de uma notação compacta para a definição de conjuntos de pinos. As entradas e saídas ativas em nível BAIXO podem ser especificadas na seção de definição dos pinos, utilizando um ponto de exclamação (!) em frente ao nome do pino. Este recurso adicional permite que as equações sejam escritas em função de quando elas estão "ativas" sem se preocupar se o nível ativo é representado por um nível ALTO ou um nível BAIXO. Alguns exemplos de declaração dos pinos feita de modo individual podem ser vistos na Tabela 12-1, onde também se pode ver a notação compacta.

TABELA 12-1 Métodos de declaração dos pinos.

Declarações Individuais	Notação Compacta
PIN 1 = CLK;	PIN 1 = CLK;
PIN 2 = !CLR;	PIN [2,3] = ![CLR,SET];
PIN 3 = !SET;	PIN [4..7] = [Q0..3];
PIN 4 = Q0;	
PIN 5 = Q1;	
PIN 6 = Q2;	
PIN 7 = Q3;	

Uma vez que todas as entradas e saídas estejam declaradas, as equações lógicas que definem a operação do circuito podem ser apresentadas. O formato geral de todas as equações é:

variável = expressão lógica

O lado esquerdo da equação pode ser qualquer nome de uma saída ou qualquer outro nome de variável ou sinal de controle que está definido no projeto. O lado direito da equação (expressão lógica) consiste em uma combinação

```
Name                    xxxxx;
Partno                  xxxxx;
Date
Revision
Designer
Company
Assembly
Location
/*****************************************************************************/
/*                                                                           */
/*                                                                           */
/*                                                                           */
/*****************************************************************************/
/*                     Dispositivo a Ser Programado                          */
/*****************************************************************************/
/*      Entradas */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
/*      Saídas   */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
pin 1   =                   ;        /*                                      */
/*****************************************************************************/
/*           Equações                                                        */
/*****************************************************************************/
```

Fig. 12-12 Um "modelo" de um arquivo de entrada para o compilador CUPL.

TABELA 12-2 Sintaxe CUPL para as operações lógicas.

Função	Operador	Formato CUPL	Formato Convencional
AND	&	A & B	A · B
OR	#	A # B	A + B
NOT	!	!A	$\overline{A}$
EX-OR	$	A $ B	A ⊕ B

lógica qualquer (AND, OR, NOT, EX-OR) das variáveis definidas no projeto. A sintaxe apropriada para este operadores pode ser vista na Tabela 12-2.

Em alguns casos, uma equação deve ser escrita para definir a operação de um ponto interno do circuito (nó) no PLD. Por exemplo, a GAL 16V8A tem um buffer tristate associado com cada saída. Nos modos de operação complexo e registrador, a habilitação da saída tristate é controlada por um termo produto da matriz de entrada. O CUPL e muitos outros sistemas de desenvolvimento fornecem esse recurso permitindo extensões no nome da variável de saída. Por exemplo, admita que um pino de saída é denominado DECODE, como pode ser visto na Fig. 12-13. A equação lógica para DECODE aciona esta saída com nível lógico 1 somente quando A está em ALTO e B está em BAIXO. A equação lógica para DECODE.OE define as condições para a habilitação da saída tristate. Neste exemplo, a saída está em alta impedância sempre que a entrada *C* estiver em nível ALTO e estará habilitada quando *C* estiver em BAIXO. Se desejássemos habilitar incondicionalmente a saída, teríamos que escrever a equação

DECODE.OE = 'b'1

Ela indica que a entrada de habilitação do buffer tristate está sempre em nível lógico 1. A notação "b" é usada para indicar ao CUPL que o valor especificado está em "binário", e não em decimal ou hexadecimal.

Um outro exemplo do uso de extensões pode ser visto na implementação de circuitos seqüenciais síncronos. Você deve se lembrar do Cap. 7 que máquinas de estado eram projetadas escrevendo-se equações para as *entradas* de um flip-flop. No modo registrador da GAL 16V8A, as equações devem ser escritas para a entrada *D* de cada flip-flop, que, por sua vez, irá determinar a saída do flip-flop na próxima transição do clock. Isto é feito utilizando a extensão .D no nome do pino de saída. Por exemplo, a equação para a saída Q_0 do flip-flop D que está conectado em modo de comutação seria:

Q0.D = !Q0

Esta equação pode ser expressa em palavras da seguinte forma: a entrada *D* do flip-flop Q_0 (isto é, Q0.D) está conectada ao inverso da saída Q_0. Outros PLDs que não a GAL 16V8A podem necessitar de outras extensões.

Ciclo de Desenvolvimento

O primeiro passo em qualquer projeto é definir cuidadosamente o problema e seu escopo. A determinação de todas as entradas e saídas necessárias é resultado deste passo. A relação entre todas as saídas e entradas também está definida e pode ser expressa de diversos modos, tais como: tabelas-verdade, tabelas de estado, equações lógicas, ou mesmo diagramas lógicos. O segundo passo consiste em criar um arquivo de entrada, também chamado de "arquivo fonte", no formato exigido pelo compilador. A seguir, o compilador é chamado e lhe é fornecido o nome do arquivo de entrada. O processo de compilação produz vários arquivos de saída diferentes. O arquivo de documentação (extensão .DOC para o CUPL) contém qualquer mensagem de erro que tenha sido gerada pelo compilador. O arquivo do mapa de fusíveis mostra os fusíveis que

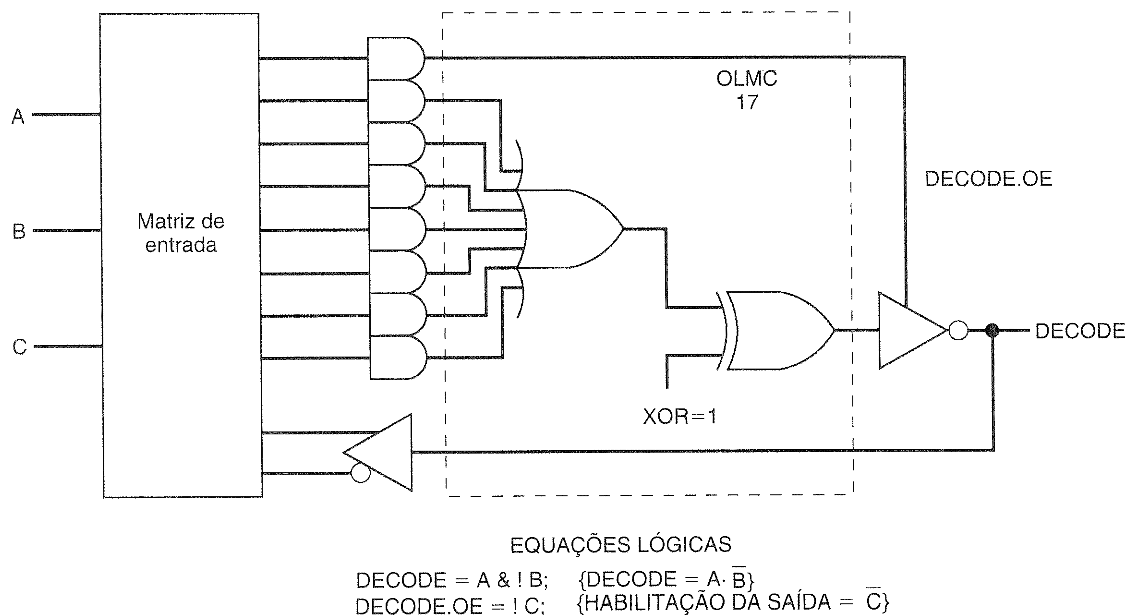


Fig. 12-13 Usando extensões para especificar o controle de habilitação da saída.

serão queimados no PLD. Este arquivo é convertido em um arquivo JEDEC para que possa ser usado por um programador.

Se o compilador gerar qualquer mensagem de erro, a causa deve ser determinada e corrigida usando um editor de textos para alterar o arquivo de entrada. Esse processo é repetido até que não haja mais erros. Neste ponto, o projeto pode ser testado usando-se um **simulador**. Um simulador é um programa que determina os estados lógicos corretos das saídas a partir da descrição do circuito lógico e dos valores atuais das entradas. É fornecido um conjunto de entradas hipotéticas que irão demonstrar que o dispositivo trabalha como esperado. Estes são chamados de **vetores de teste**. Se os vetores de teste forem suficientemente detalhados, o projeto pode ser experimentado antes que o primeiro dispositivo seja programado. Quando o projetista estiver convencido de que o projeto vai funcionar, o arquivo JEDEC é gerado e o software de programação é chamado. O arquivo JEDEC serve como arquivo de entrada para o software de programação, e o PLD é colocado no programador. Muitos programadores e seus respectivos software são capazes não só de programar o PLD mas também de fornecer os vetores de teste e monitorar as saídas. Esta verificação nos diz que o PLD está funcionando corretamente. O PLD é então colocado em um circuito e é testada a sua funcionalidade, juntamente com os outros componentes do circuito. O fluxograma da Fig. 12-14 mostra o processo de desenvolvimento completo.

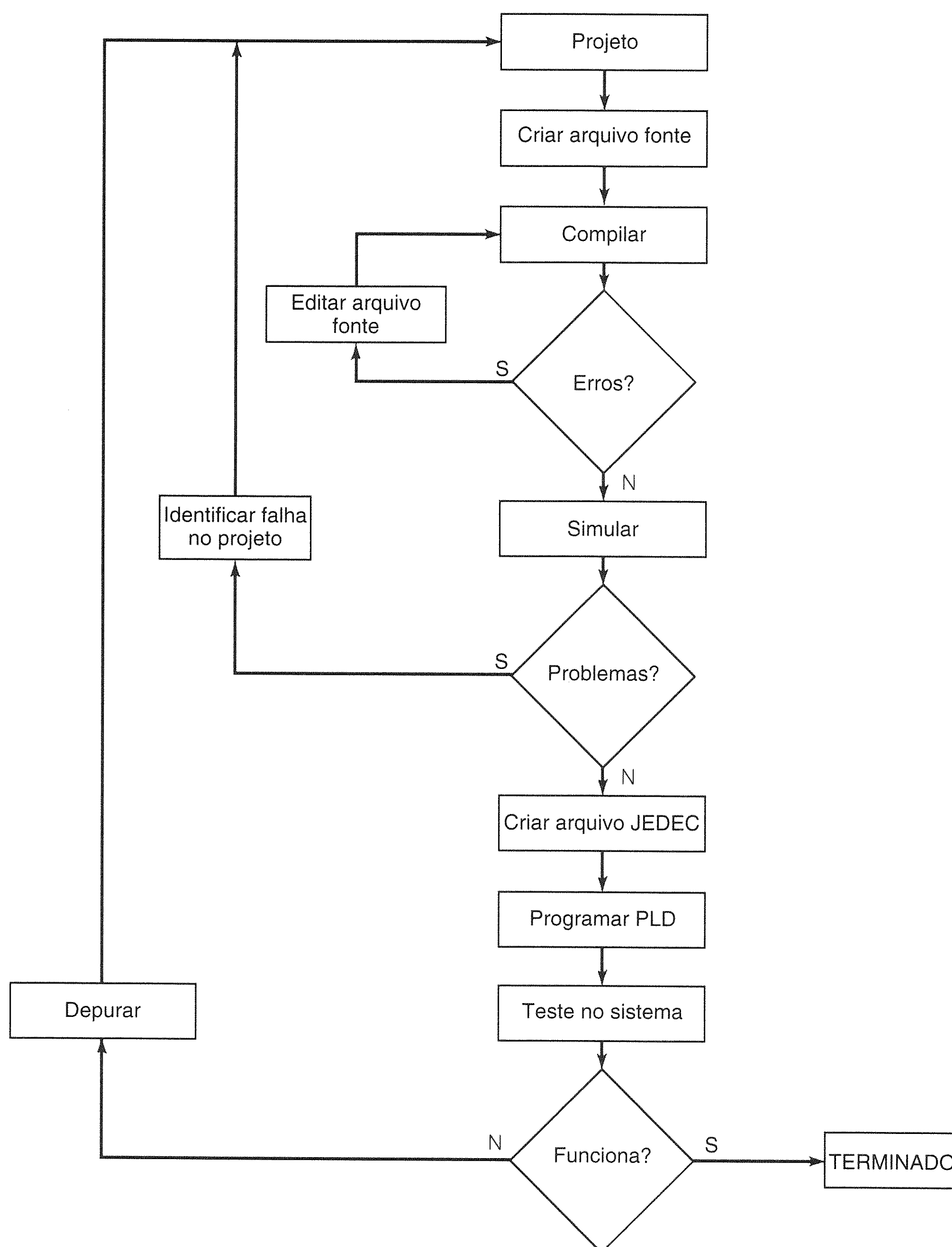


Fig. 12-14 Fluxograma do ciclo de desenvolvimento com PLDs.

Exemplos

Um exemplo simples usando o CUPL servirá para ilustrar o processo. Veja a tabela-verdade da Tabela 12-3, que representa o resultado da descrição de um problema e da especificação do escopo para um determinado projeto. Se estivéssemos usando chips lógicos SSI, o próximo passo seria a simplificação usando mapas de Karnaugh. Geralmente não é necessário simplificar o projeto quando estamos usando PLDs, uma vez que é o compilador que vai realizar a simplificação. Entretanto, equações na forma de soma de produtos devem ser escritas a partir da tabela-verdade.

O próximo passo é criar um arquivo fonte (de entrada) usando um editor de textos. O cabeçalho do arquivo fonte deve ser preenchido para fornecer a documentação completa, como se pode ver na Fig. 12-15. A primeira grande decisão que deve ser tomada é a seleção dos pinos de entrada e saída. Isto requer alguma compreensão do funcionamento da arquitetura do dispositivo que está sendo programado. Neste simples projeto usando a GAL 16V8A, praticamente quaisquer pinos que não V_{CC} e terra poderiam ter sido usados como entradas. O pino de saída deve ser escolhido entre os pinos 12-19. Finalmente, escreve-se uma expressão na forma de soma de produtos para especificar a operação do pino de saída. Observe os símbolos que foram usados para representar as operações lógicas.

O arquivo fonte mostrado na Fig. 12-15 está em um formato muito simples e pode ser utilizado com a maioria das ferramentas de desenvolvimento, desde que sejam feitas algumas pequenas modificações. O compilador, na verdade, simplifica as equações e gera um arquivo de documentação que mostra as equações simplificadas e o mapa de

TABELA 12-3 Tabela-verdade de uma lógica combinacional.

A	B	C	D	X	
0	0	0	0	0	
0	0	0	1	1	→ $\bar{A}\bar{B}\bar{C}D$
0	0	1	0	0	
0	0	1	1	0	
0	1	0	0	0	
0	1	0	1	1	→ $\bar{A}B\bar{C}D$
0	1	1	0	0	
0	1	1	1	0	
1	0	0	0	0	
1	0	0	1	0	
1	0	1	0	0	
1	0	1	1	0	
1	1	0	0	0	
1	1	0	1	1	→ $AB\bar{C}D$
1	1	1	0	0	
1	1	1	1	1	→ $ABCD$

$$\left\{ X = \bar{A}\bar{B}\bar{C}D + \bar{A}B\bar{C}D + AB\bar{C}D + ABCD \right\}$$

```
Name                    combo.pld;
Partno                  12-17;
Date                    01/06;
Revision                01;
Designer                N. Widmer;
Company                 Purdue University;
Assembly                Cap. 12;
Location                Livro Tocci;
Device                  G16V8A;
Format                  j;          /*  JEDEC        */
/**********************************************************************/
/*   Este é um exemplo de lógica combinacional com 4 entradas e uma saída  */
/*                                                                    */
/*                                                                    */
/**********************************************************************/
/*             Dispositivo a Ser Programado Gal 16V8A                 */
/**********************************************************************/

/*  Entradas     */

pin 1   = A              ;      /*Observe o uso do PINO 1 como entrada*/
pin 2   = B              ;      /*B, C e D são entradas normais       */
pin 3   = C              ;      /*                                    */
pin 4   = D              ;      /*                                    */

/*    Saídas     */

pin 19  = X              ;      /*Pino 12 é a saída                   */

/**********************************************************************/
/*                 Equações                                           */
/**********************************************************************/

X = !A&!B&!C&D # !A&B&!C&D # A&B&!C&D # A&B&C&D;
```

Fig. 12-15 Arquivo de entrada para o circuito combinacional.

```
*****************************************************************************
                              combo.pld
*****************************************************************************
=============================================================================
                        Expanded Product Terms
=============================================================================

X =>
    !A & !C & D
  # A & B & D

=============================================================================
                              Fuse Plot
=============================================================================

Syn   02192 - Ac0   02193 x

Pin #19  02048  Pol -  02120  Ac1 x
 00000 ---x-x--x-----------------------
 00032 x-x-----x-----------------------
 00064 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00096 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00128 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00160 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00192 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00224 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Fig. 12-16 Arquivo de documentação com mapa de fusíveis.

fusíveis para a programação do dispositivo. Partes desse arquivo de documentação para esse projeto podem ser vistas na Fig. 12-16. Observe que a expressão simplificada é igual à que teria sido obtida com o mapa de Karnaugh. O mapa de fusíveis mostra as conexões na matriz de entrada, identificando cada bit programável por um endereço. Cada posição marcada com um "x" indica uma conexão da linha com a coluna da matriz (fusível não deve ser aberto). Cada posição marcada com um "–" indica um fusível aberto e que, portanto, não há conexão entre a linha e a coluna. A Fig. 12-17 mostra as primeiras duas colunas do mapa de fusíveis superposto ao diagrama interno da GAL 16V8A para

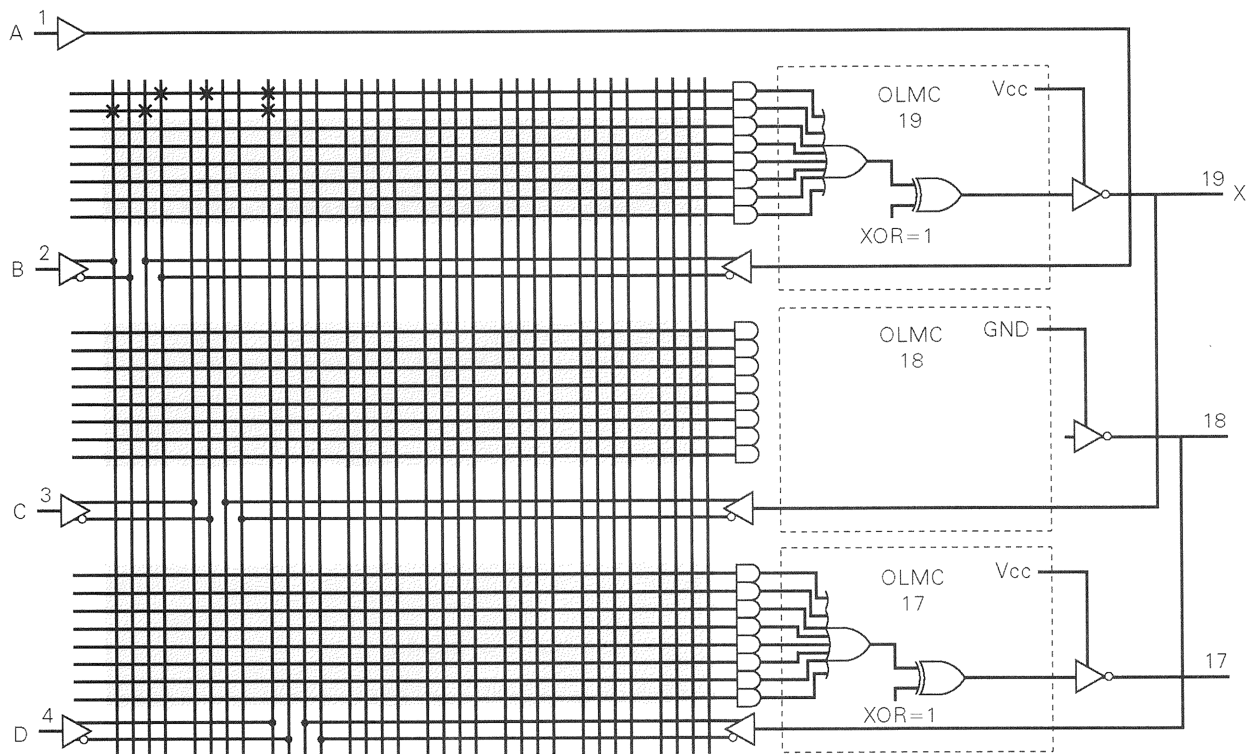


Fig. 12-17 Implementação do circuito em um PLD.

este projeto. Os bits de seleção de modo (SYN = 1, AC0 = 0) estão configurados para o modo simples, e AC1 = 0 para a OLMC 19 faz com que a saída seja combinacional.

O exemplo anterior é bastante trivial e serve apenas para demonstrar os detalhes do desenvolvimento de um projeto usando PLD. Este circuito poderia ser implementado utilizando portas lógicas convencionais, com um menor custo e requisitos de potência e espaço em placas semelhantes. A grande vantagem no uso de PLDs aparece quando implementamos circuitos que não possuem alternativas convencionais ou que necessitam de um grande número de chips convencionais. O próximo exemplo mostra o projeto de um circuito convencional que necessita de uma combinação "monstruosa" de portas TTL para ser implementada. Este circuito é bastante funcional e ainda assim pode ser implementado em uma única GAL 16V8A a um custo razoável.

EXEMPLO 12-4

Projete um circuito que indica qual dos concorrentes em um concurso de perguntas e respostas deve responder primeiro à pergunta. O circuito deve ter como entradas um botão de contato *momentâneo* para cada um dos seis concorren-

```
Name                 GAME.PLD;
Partno               12-20;
Date                 01/06;
Revision             01;
Designer             N. Widmer;
Company              Purdue University;
Assembly             Cap. 12;
Location             Livro Tocci;
Device               G16V8A;
Format               j;        /*      JEDEC    */
/*************************************************************************/
/* ESTE DISPOSITIVO DETERMINA QUE O BOTÃO FOI APERTADO PRIMEIRO E É      */
/* USADO EM CONCURSO DE PERGUNTAS E RESPOSTAS                            */
/*                                                                       */
/*************************************************************************/
/*    Dispositivo a Ser Programado GAL 16V8A                             */
/*************************************************************************/
/*     Entradas */

pin [2..7]   = [S1..6]    /*Estas seis entradas (S1-S6)             */
                          /*são usadas por seis jogadores.          */
                          /*Elas estariam conectadas a botões de    */
                          /*contato momentâneo, normalmente em nível */
                          /*BAIXO.                                  */
                          /*                                        */

pin 9    = RESET  ;       /*Este botão  é
                            usado pelo animador
                            para apagar os LEDs.                    */

/*       Saídas  */

pin 12  = !L1             ;       /* Estas saídas vão para os LEDs     */
pin 13  = !L2             ;       /* que indicam que jogador           */
pin 14  = !L3             ;       /* deve responder primeiro.          */


pin 17  = !L4             ;       /*Observe que eles são               */
pin 18  = !L5             ;       /*ativos em nível BAIXO pelo uso de !*/
pin 19  = !L6             ;       /*na definição da saída.             */
/*************************************************************************/
/*              Equações                                                 */
/*************************************************************************/

L1 = !RESET & S1 &      !L2 & !L3 & !L4 & !L5 & !L6  #  !RESET & L1;

L2 = !RESET & S2 & !L1 &      !L3 & !L4 & !L5 & !L6  #  !RESET & L2;

L3 = !RESET & S3 & !L1 & !L2 &      !L4 & !L5 & !L6  #  !RESET & L3;

L4 = !RESET & S4 & !L1 & !L2 & !L3 &      !L5 & !L6  #  !RESET & L4;

L5 = !RESET & S5 & !L1 & !L2 & !L3 & !L4 &      !L6  #  !RESET & L5;

L6 = !RESET & S6 & !L1 & !L2 & !L3 & !L4 & !L5       #  !RESET & L6;
```

Fig. 12-18 O arquivo de entrada do circuito do Exemplo 12-4.

tes e seis LEDs para indicar qual dos concorrentes apertou o botão primeiro. Um botão de reset é utilizado pelo animador do concurso para apagar todas as luzes (LEDs).

Solução

Este problema possui elementos de lógica combinacional, bem como de lógica seqüencial. Considere as condições sob as quais um LED que representa um determinado concorrente deve ser aceso. O LED *N* deve ser aceso se ocorrer uma das condições citadas a seguir:

1. O concorrente *N* está apertando o botão
   E nenhum outro concorrente está com seu LED aceso
   E o animador não está tentando apagar os LEDs.

OU

2. O concorrente *N* está com o LED aceso
   E o animador não está tentando apagar os LEDs.

A primeira condição especifica uma combinação de entradas para acender o LED. A segunda condição gera uma ação de armazenamento (seqüencial) para manter o LED aceso após todos os botões terem sido liberados.

O primeiro passo para implementar uma solução baseada em PLD é dar nome a todas as entradas e saídas e especificar os níveis lógicos ativos. Os botões 1-6, denominados S1–S6, são ativos em nível ALTO. A entrada de RESET também é ativa em ALTO. As saídas são denominadas L1–L6 e devem ser ativas em nível BAIXO para acender os LEDs, uma vez que as saídas podem absorver mais corrente do que podem fornecer (vide Cap. 8). Usando esses nomes, as afirmações acima podem ser escritas da seguinte forma para a saída L1:

$$L1 = \underbrace{S1 \cdot \overline{L2} \cdot \overline{L3} \cdot \overline{L4} \cdot \overline{L5} \cdot \overline{L6} \cdot \overline{RESET}}_{\text{Condição 1}} + \underbrace{L1 \cdot \overline{RESET}}_{\text{Condição 2}}$$

O arquivo de entrada para o CUPL pode ser visto na Fig. 12-18. As entradas estão declaradas usando a notação de conjunto (compacta) do CUPL. Qualquer um dos pinos de entrada (1-9, 11) poderia ter sido utilizado. Os pinos de saída foram selecionados conforme mostrado porque estamos operando no modo simples. Neste modo, os pinos 15 e 16 não podem ser realimentados, e portanto eles devem ser evitados. Observe que, para especificar uma saída como ativa em BAIXO, o operador de inversão (!) é usado antes da definição da declaração do nome do pino. Isto diz ao compilador para fazer com que a saída vá para BAIXO quando ela estiver ativa, enquanto a expressão lógica indica que é necessária lógica positiva para tornar a saída ativa. O CUPL automaticamente se encarrega de complementar as saídas e qualquer termo de realimentação que seja necessário.

Imagine tentar implementar esse circuito usando lógica TTL! Ele pode ser completamente implementado usando uma única GAL 16V8A.

Circuitos com saídas síncronas também podem ser implementados usando a GAL 16V8A. No Cap. 7, estudamos o projeto de contadores síncronos com flip-flops JK. Todos os estados atuais possíveis foram relacionados em uma tabela, juntamente com seus "próximos estados". Então, determinamos as entradas necessárias para fazer com que o estado atual fosse para o próximo estado desejado, resultando em equações lógicas para cada entrada *J* e *K*.

O mesmo procedimento é usado quando projetamos circuitos síncronos com PLDs. Entretanto, a GAL 16V8A possui flip-flops D em vez de flip-flops JK. Isso na verdade faz com que o processo se torne mais simples, uma vez que você precisa escrever apenas uma equação para cada entrada *D*. A equação para cada entrada *D* é simplesmente a soma de todos os estados presentes (termos produto) que devem fazer com que o flip-flop vá para ALTO no próximo estado.

EXEMPLO 12-5

Para ilustrar esse processo, implementamos o contador síncrono da Seção 7-14 (Fig. 7-33) usando PLDs. O primeiro passo ainda é criar uma tabela relacionando o estado atual com o próximo estado, conforme foi visto no passo 3 do procedimento de projeto de circuitos síncronos, na Seção 7-14. Você pode pensar essa tabela como sendo um conjunto de três tabelas-verdade, cada uma especificando o próximo estado de cada flip-flop com base no estado atual de todas as entradas. Para essa tabela, as expressões na forma de soma de produtos não-simplificadas para o próximo estado de cada saída são:

$$\text{Próximo estado de C} = \overline{C} \cdot B \cdot A$$
$$\text{Próximo estado de B} = \overline{C} \cdot \overline{B} \cdot A + \overline{C} \cdot B \cdot \overline{A}$$
$$\text{Próximo estado de C} = \overline{C} \cdot \overline{B} \cdot \overline{A} + \overline{C} \cdot B \cdot \overline{A}$$

Uma vez que o próximo estado de um flip-flop D é determinado pelo estado atual em sua entrada *D*, estas também são as equações necessárias para excitar as entradas *D*. O arquivo de entrada resultante pode ser visto na Fig. 12-19. Lembre-se de que C.D é a notação usada pelo CUPL para a entrada *D* do flip-flop *C*, e também usada para os flip-flops A e B (A.D e B.D).

```
Name                      Counter.PLD;
Partno                    12-21;
Date                      01/06;
Revision                  01;
Designer                  N. Widmer;
Company                   Purdue University;
Assembly                  Cap. 12;
Location                  Livro Tocci;
Device                    G16V8A;
Format                    j;
/*************************************************************************/
/*  Este dispositivo vai funcionar como um contador de módulo 5.         */
/*  0 > 1 > 2 > 3 > 4 > 0    Todos os outros estados vão para 0.         */
/*************************************************************************/
/*           Dispositivo a Ser Programado GAL 16V8A                      */
/*************************************************************************/
/*   Entradas   */

PIN 1 = CK;              /*entrada de clock (tem que ser o pino 1)*/

/*     Não há entradas combinacionais para este contador de módulo 5     */

/*     Saídas   */

pin 19  = C                  ;        /*   Saída mais significativa    */
pin 18  = B                  ;        /*                               */
pin 17  = A                  ;        /*   Saída menos significativa   */

/*************************************************************************/
/*            Equações                                                   */
/*************************************************************************/

C.D = !C & B & A;

B.D = !C & !B & A  #  !C & B & !A;

A.D = !C & !B & !A #  !C & B & !A;
```

Fig. 12-19 Arquivo de entrada para o contador de módulo 5.

Questões de Revisão

1. Para que os caracteres /* */ são usados em um arquivo de entrada?
2. Qual é o significado de usar .OE em um nome de uma saída?
3. Explique o que representa QA.D.
4. Defina *arquivo fonte*.
5. Qual o arquivo de saída que identifica erros de compilação?
6. Em qual arquivo você corrige os erros de compilação?

12-5 COMENTÁRIOS FINAIS

Neste capítulo, fornecemos uma descrição detalhada de um determinado tipo de dispositivo de lógica programável, a GAL 16V8A. Embora essa descrição seja adequada para nos iniciarmos no uso de PLDs, ela não fornece uma ampla cobertura nem sobre o CUPL e nem sobre a GAL 16V8A. Uma melhor compreensão do assunto pode ser obtida utilizando-se as ferramentas de desenvolvimento enquanto se experimenta com PLDs.

O compilador CUPL é bastante versátil, e praticamente qualquer dispositivo pode ser programado pelo pacote de software fornecido pela Logical Devices. A sintaxe que você aprendeu neste capítulo se aplica a muitos outros tipos de dispositivos que podem ser programados pelo CUPL.

À medida que vamos apreciando as vantagens desse PLD em particular sobre os chips SSI e MSI, devemos lembrar que este dispositivo (GAL 16V8A) representa apenas um dos dispositivos de lógica programável mais simples. Outros dispositivos, maiores e mais poderosos, estão disponíveis. Um PLD muito comum e que é amplamente utilizado hoje é a 22V10. Vários fabricantes oferecem este dispositivo, e ele está disponível em versões reprogramáveis ou não. Ele possui mais entradas e mais OLMCs. Entretanto, sua arquitetura é bastante similar à da GAL 16V8A.

Uma nova linha de componentes foi desenvolvida pela Lattice, e eles são "programáveis no sistema". A ISPGAL22V10 é o dispositivo mais simples desta linha. A grande vantagem de componentes ISP (*In-System Programable* — programáveis no sistema) é que eles não precisam de um programador: basta um cabo para conexão do dispositivo à porta paralela de um PC. A única desvantagem desses dispositivos é que eles estão disponíveis apenas em encapsulamentos do tipo PLCC. Isto significa que seriam necessárias modificações nas práticas de laboratório que envolves-

TABELA 12-4 Fabricantes e revendas de ferramentas de desenvolvimento com PLDs.

Software e Programadores	Software	Programadores
CUPL® Logical Devices, Inc. 130 Capital Dr. Suite A Golden, CO 80401 1-800-331-7766 http://logicaldevices.com ABEL®/Synario 1-888-796-2746 Data I/O® Corp. 1-800-332-8246 10525 Willows Rd. NE Redmond, WA 98052 http://data-io.com	Minc, Inc. 6755 Earl Dr. Colorado Springs, CO 80918 1-719-590-1155 http://www.minc.com ORCAD Systems, Inc. 9300 SW Nimbus Ave. Beaverton, OR 97008 1-503-671-9500 http://www.orcad.com	EMP-20 Needham's Electronics 1-916-924-8037 Disponível através da Digi-Key 1-800-344-4539 http://www.digikey.com

sem protoboards. Entretanto, a possibilidade de reprogramar o dispositivo após ele ter sido colocado no sistema abre caminho para uma nova abordagem no projeto de sistemas digitais com a possibilidade de termos hardware reconfigurável. Por exemplo, um sistema baseado em um microprocessador seria capaz de determinar quais as opções estão atualmente instaladas e então reconfigurar a interface lógica de acordo. Existem ainda outras linhas de componentes com características especiais, como por exemplo saídas com alta capacidade de corrente.

O campo dos dispositivos de lógica programável vai continuar crescendo. Estes dispositivos irão ocupar o espaço dos circuitos lógicos SSI e MSI, que têm nos servido tão bem nos últimos 25 anos. Este capítulo forneceu uma base a partir da qual você poderá evoluir nessa área excitante e de contínuo desenvolvimento. A Tabela 12-4 relaciona algumas empresas que fornecem materiais de que você precisará para trabalhar nessa área.

RESUMO

1. Os dispositivos de lógica programável (PLDs) serão de grande importância no futuro dos sistemas digitais.
2. Os PLDs reduzem a quantidade de peças necessárias em estoque, simplificam o protótipo de circuitos, reduzem o tamanho e a potência necessária de um produto e permitem que este seja facilmente atualizado.
3. A GAL 16V8A é um dos PLDs disponíveis mais simples, mas ainda é bastante utilizado e demonstra os princípios básicos que estão por trás de todos os PLDs.
4. A GAL 16V8A tem uma matriz de conexões programáveis que são usadas para selecionar os termos de entrada para um grupo de portas AND. É feita uma operação OR com as saídas destas portas AND em uma macrocélula da lógica de saída (OLMC).
5. A OLMC também pode ser programada para complementar o valor do bit resultante, armazenar este bit em um flip-flop D, utilizar um buffer de saída tristate ou realimentá-lo para a matriz de entrada. A combinação desejada desses recursos é selecionada programando-se o componente em um dos três modos de operação e escolhendo-se uma das configurações possíveis para o modo programado.
6. No modo simples, cada OLMC pode ter oito termos produtos. Nenhuma das saídas pode ser do tipo tristate, e os flip-flops não são usados para armazenar a saída. Todos os pinos, exceto o 15 e 16, podem ser realimentados.
7. No modo complexo, cada OLMC pode ter sete termos produtos. As saídas tristate estão habilitadas por uma equação lógica à parte. Todos os pinos, exceto o 12 e 19, podem ser realimentados.
8. No modo registrador, cada célula pode ter oito termos produtos de entrada. O bit resultante é armazenado no flip-flop D. O sinal de clock e o de habilitação de saída para estas células estão nos pinos 1 e 11, respectivamente. As células que estão configuradas como combinacionais funcionam do mesmo modo descrito para o modo complexo.
9. Para utilizar PLDs, você precisa de um sistema de desenvolvimento que consiste em um computador, um programa para desenvolvimento de projetos com PLDs e um programador com um software para acioná-lo.
10. As equações lógicas estão descritas no arquivo de entrada em um formato que é ditado pelo software de desenvolvimento. Ele traduz as equações lógicas para o formato JEDEC que pode ser usado pelo programador. O arquivo JEDEC é enviado para o programador, e o componente é programado.
11. Os PLDs são fáceis de aprender e usar, e estão disponíveis com recursos que excedem em muito os da GAL 16V8A.

TERMOS IMPORTANTES

matriz de entrada
macrocélula da lógica de saída
downloading
soquete ZIF
JEDEC
compilador
simulador
vetor de teste

PROBLEMAS

SEÇÃO 12-1

12-1. Faça uma tabela relacionando cada modo e configuração para a OLMC de uma GAL juntamente com seus bits de controle (SYN, AC0, AC1). Identifique quais OLMCs podem ser usadas em cada configuração.

12-2. Escreva todos os modos que podem ser usados para implementar os recursos a seguir:
(a) Um latch *D* com saídas tristate
(b) Um flip-flop D
(c) Um contador síncrono
(d) Um circuito decodificador

12-3. Para cada modo e configuração, diga qual é o número máximo de termos em uma expressão do tipo soma de produtos.

12-4. Identifique a fonte do sinal de realimentação (*FMUX*) para cada modo e configuração.

12-5. Identifique a fonte do sinal de habilitação das saídas tristate (*TSMUX*) para cada modo e configuração.

12-6. Mostre que, programando o bit XOR em nível ALTO, o nível do pino de saída será o mesmo que o da saída da porta OR na OLMC para os Exemplos 12-1, 12-2 e 12-3.

SEÇÃO 12-2

12-7. Relacione os equipamentos necessários para desenvolver circuitos usando PLDs.

SEÇÃO 12-4

12-8. Escreva um arquivo de entrada que programaria uma GAL 16V8A para funcionar como um decodificador 74LS138 de 3 linhas para 8 linhas.

12-9. Desenhe o diagrama completo para o circuito do Exemplo 12-4 se este fosse implementado usando chips convencionais.

12-10. Escreva um arquivo de entrada para o CUPL que programaria uma GAL 16V8A para funcionar como um contador binário de módulo 8.

12-11. Um motor de passo pode ser movimentado energizando-se seus enrolamentos na seguinte seqüência:

1 0 1 0
1 0 0 1
0 1 0 1
0 1 1 0

Escreva as equações no formato CUPL que irão criar o contador que poderá acionar o motor de passo.

C
12-12. Passando pela seqüência de estados do Problema 12-11 na direção oposta, o motor de passo vai girar na direção oposta. Escreva as equações que recebam uma entrada externa (dir) e que façam girar o motor no sentido horário quando dir = 0 e no sentido anti-horário quando dir = 1 (o motor avança um passo na transição do clock). Escreva o arquivo de entrada para programar a GAL 16V8A.

C
12-13. Projete uma fechadura eletrônica. Seu circuito deve ter como entrada uma combinação de três bits e um botão ENTRA. Se a seqüência correta de valores de três bits é fornecida (101 <ENTRA>, 100 <ENTRA>, 111 <ENTRA>), o bit de saída deve ir para ALTO, e a porta deve ser destrancada.

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 12-1

1. Apagável e reprogramável
2. Simples, complexo e registrador
3. Veja o texto
4. Registrador/configuração síncrona
5. Registrador

SEÇÃO 12-2

1. Software de desenvolvimento; software de programação
2. **(a)** Um arquivo produzido pelo software de desenvolvimento que mostra que fusíveis devem ser queimados **(b)** A mesma informação é traduzida em um formato apropriado para transmissão para o programador
3. Um arquivo de saída (mapa de fusíveis) de acordo com o padrão JEDEC

SEÇÃO 12-3

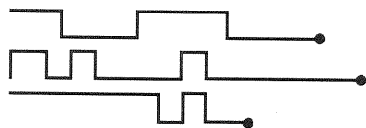
1. Um compilador tem várias opções para a entrada de dados sobre o projeto de um circuito.
2. Arquivo ASCII; captura de esquemático; tabelas-verdade; tabelas de estado; diagramas de tempo
3. Linguagem de descrição de hardware VHSIC

SEÇÃO 12-4

1. Para delimitar comentários.
2. Para indicar que este sinal aciona a habilitação da saída
3. A entrada *D* do flip-flop Q_A
4. O arquivo de entrada que especifica as relações entre as entradas e saídas
5. Arquivo de documentação
6. Fonte (arquivo de entrada)

CAPÍTULO 13

Introdução ao Microprocessador e ao Microcomputador



SUMÁRIO

■ OBJETIVOS

Ao completar este capítulo, você deverá estar apto a:

- Descrever a função e a operação de cada um dos cinco elementos básicos de qualquer computador.
- Compreender que um computador repete continuamente uma seqüência de operações de busca e execução.
- Entender a diferença entre um microprocessador e um microcomputador.
- Analisar os ciclos de busca e execução durante a execução de um programa em linguagem de máquina.
- Compreender a função dos diferentes tipos de barramentos e seus sinais em um microcomputador.
- Citar as principais funções realizadas por um microprocessador.
- Descrever os diferentes tipos de palavras utilizadas nos computadores.
- Atualizar seu conhecimento aprendendo mais detalhes e conceitos avançados sobre sistemas baseados em microprocessadores.

■ INTRODUÇÃO

Não é exagero dizer que o microprocessador e o microcomputador revolucionaram a indústria eletrônica e tiveram um enorme impacto em diversos aspectos de nossas vidas. O desenvolvimento de CIs de altíssima densidade reduziu tão drasticamente o tamanho e o custo dos microcomputadores que os projetistas rotineiramente consideram utilizar suas capacidades e versatilidade em uma grande variedade de produtos e aplicações.

Neste capítulo vamos estudar os princípios básicos de operação de um microcomputador. Apesar de focarmos no microcomputador, a maioria dos conceitos e idéias se aplica aos computadores de todos os portes. Para ilustrar de maneira representativa os diversos aspectos operacionais dos microcomputadores, utilizamos um microprocessador específico, o Intel 8051.

A família 8051 representa um ramo da árvore da família dos microprocessadores. Conforme a tecnologia de microprocessadores e microcomputadores evoluiu desde as origens de 4 bits (Intel 4004), no início dos anos 1970, para os processadores de 8 bits (8008, 8080, 8085, 6800, 6502, Z80 etc.) no final dos anos 1970, as aplicações se dividiram em dois ramos distintos. Um destes ramos foi a infância dos computadores pessoais. Estes computadores baseados em microprocessadores pretendiam ser ferramentas versáteis que poderiam carregar e executar vários programas, tais como processadores de texto, planilhas, banco de dados e jogos. Eles poderiam ser facilmente programados pelo usuário para realizar o que o programador imaginasse. Os descendentes de 16 e 32 bits (8086, as séries 80x86 e 680x0) simplesmente evoluíram daí. A arquitetura básica do microprocessador não mudou substancialmente desde os primeiros sistemas, embora a velocidade e a quantidade de memória endereçada diretamente tenham aumentado bastante.

O segundo ramo que se desenvolveu a partir de 1970 foi a utilização de computadores baseados em microprocessadores como unidade de controle embutida num produto comercial. Este tipo de microcomputador é construído com os mesmos elementos de um computador pessoal, mas é programado apenas uma vez pelo fabricante. Deste modo, ele passa sua "vida" realizando suas tarefas específicas, tais como esperar que teclas sejam pressionadas e ligar e desligar dispositivos tais como luzes, motores, campainhas etc. A família 8051 se enquadra nesta categoria geral conhecida como microcontroladores *dedicados*. Esta família de dispositivos foi desenvolvida no início dos anos 1980 juntamente com o 68HC11, mas as aplicações desses dispositivos de oito bits orientados para controle se tornaram tão difundidas que eles ainda são muito utilizados hoje em dia. Muitas versões diferentes do 8051 original foram desenvolvidas com características especiais, mas elas ainda utilizam o mesmo conjunto básico de instruções e a arquitetura do original. Muitos outros microcontroladores, mais complexos e de maior poder de processamento, foram desenvolvidos para aplicações que exigem mais do que a capacidade de um computador de oito bits.

A família 8051 foi escolhida porque é muito utilizada atualmente, e além disso é capaz de demonstrar como os computadores funcionam de um modo adequado para um estudante iniciante. Assim que você possuir uma compreensão básica de um dos primeiros microprocessadores, tal como o 8051, é uma tarefa mais fácil evoluir para os chips mais avançados.

Seria necessário um livro completo para cobrir todos os detalhes importantes da operação de computadores e microprocessadores. O material neste capítulo pretende apenas fornecer uma base para estudos posteriores.

13-1 O QUE É UM COMPUTADOR DIGITAL?

Um computador digital é uma combinação de circuitos e dispositivos digitais que podem realizar uma seqüência *programada* de operações com mínima intervenção humana. A seqüência de operações é chamada de **programa**. O programa é um conjunto de instruções codificadas que é armazenado na memória interna do computador juntamente com todos os dados de que o programa necessita. Quando o computador é comandado a executar o programa, ele executa as instruções na ordem em que foram armazenadas na memória até que o programa termine. Ele faz isso numa velocidade extremamente alta.

13-2 COMO OS COMPUTADORES "PENSAM"?

Computadores não pensam! O **programador** do computador fornece um *programa* de instruções e dados que especificam cada detalhe sobre o que fazer e quando deve fazer. O computador é simplesmente uma máquina de alta

velocidade que pode manipular dados, resolver problemas e tomar decisões, sempre sob o controle de um programa. Se o programador cometer um erro no programa ou entrar com os dados errados, o computador produzirá resultados incorretos. Um ditado popular no campo da computação diz "lixo na entrada/lixo na saída" ("*garbage in/garbage out*").

Talvez a melhor questão a formular neste ponto é: Como um computador executa um programa de instruções? Geralmente, esta pergunta é respondida mostrando-se um diagrama de uma arquitetura computacional (organização de seus vários elementos) e então realizando-se passo a passo o procedimento que o computador realiza na execução do programa. Nós faremos isto, mas não agora. Primeiro, vamos procurar uma analogia um tanto artificial que contenha muitos dos conceitos relacionados com a operação dos computadores.

13-3 AGENTE SECRETO 89

O Agente Secreto 89 está tentando descobrir quantos dias antecedem o assassinato de um certo líder mundial. Seu contato lhe revelou que esta informação está localizada numa série de caixas postais. Para garantir que ninguém mais obtenha a informação, ela é espalhada por dez caixas diferentes. Seu contato forneceu a ele dez chaves juntamente com o seguinte conjunto de instruções:

1. A informação em cada uma das caixas está escrita em código.
2. Abra a caixa 1 primeiro e execute a instrução localizada lá.
3. Continue pelas caixas restantes em seqüência, a menos que seja instruído do contrário.
4. Uma das caixas está preparada para explodir quando for aberta.

O Agente 89 pega as dez chaves e se dirige ao correio com o livro de código nas mãos.

A Fig. 13-1 mostra os conteúdos das dez caixas após terem sido decifradas. Suponha que você seja o Agente 89; comece na caixa 1 e prossiga pela seqüência de operações para encontrar o número de dias que antecedem a tentativa de assassinato. Naturalmente, não deve ser tanto trabalho para você como seria para o Agente 89, pois você não precisa decifrar as mensagens codificadas. A resposta está no próximo parágrafo.

Se você proceder corretamente, deve terminar na caixa 6 com uma resposta de 17. Se cometeu um erro, você pode ter aberto a caixa 7, e neste caso você não estaria mais conosco. Conforme percorreu a seqüência de operações, você essencialmente duplicou os tipos de operação e encontrou muitos dos conceitos que são parte de um computador. Vamos discutir agora estas operações e conceitos no contexto da analogia do agente secreto e ver como eles estão relacionados com os computadores reais.

No caso de você não ter adivinhado ainda, as caixas postais são similares à **memória** em um computador, onde **instruções** e **dados** são armazenados. As caixas postais de 1 a 6 contêm instruções a serem executadas pelo agente secreto, e as caixas 8, 9 e 10 contêm os dados referenciados pelas instruções. (O conteúdo da caixa 7, segundo nosso conhecimento, não tem equivalente nos computadores.) Os números de cada caixa são como os **endereços** das posições de memória.

① Some o número armazenado na caixa ⑨ ao seu número de código de agente secreto.	② Divida o resultado anterior pelo número armazenado na caixa ⑩.
③ Subtraia o número armazenado na caixa ⑧.	④ Se o resultado anterior não for igual a 30, vá para a caixa ⑦. Caso contrário, continue para a próxima caixa.
⑤ Subtraia 13 do resultado anterior.	⑥ Retorne para o quartel-general para receber mais instruções.
⑦ BOMBA!	⑧ 20
⑨ 11	⑩ 2

Fig. 13-1 Dez caixas postais com mensagens codificadas para o Agente 89.

Três classes diferentes de instruções são apresentadas nas caixas de 1 a 6. As caixas 1, 2, 3 e 5 são instruções que demandam *operações aritméticas*. A caixa 4 contém uma instrução de *decisão* denominada *salto condicional* ou *ramificação condicional*. Esta instrução solicita que o agente (ou computador) decida se deve saltar para o endereço 7 ou prosseguir para o endereço 5, dependendo do resultado da operação aritmética anterior. A caixa 6 contém uma instrução de controle simples que não requer dados e não se refere a nenhum outro endereço (número de caixa). Esta instrução de *retorno* informa ao agente que o procedimento está terminado (o programa acabou) e ele não deve ir adiante. Conforme você estudar microprocessadores mais profundamente, você descobrirá como ele sabe para onde deve retornar.

Cada uma das instruções aritméticas ou de salto condicional consiste em duas partes: uma **operação** e um **endereço**. Por exemplo, a primeira parte da primeira instrução especifica a operação de adição. A segunda parte fornece o endereço (caixa 9) do dado a ser utilizado na adição. Estes dados usualmente são chamados de **operandos**, e seus endereços, de **endereços do operando**. A instrução na caixa 5 é um caso especial no qual nenhum endereço de operando é especificado. Em vez disso, o operando (dado) a ser usado na operação de subtração está incluído como parte da instrução.

Um computador, como o agente secreto, decodifica e então executa as instruções armazenadas *seqüencialmente* na memória, começando na primeira posição. As instruções

são executadas na ordem, a não ser que algum tipo de instrução de *desvio* (tal como na caixa 4) provoque um desvio ou salto na operação para um novo endereço para obter a próxima instrução. Após o desvio, as instruções são executadas seqüencialmente a partir do novo endereço.

Essas são as informações que podemos obter da analogia com o agente secreto. Cada um dos conceitos encontrados será revisto novamente no material apresentado adiante. Esperamos que a analogia tenha fornecido noções preliminares que serão úteis à medida que você inicia um estudo mais técnico sobre os computadores.

13-4 ORGANIZAÇÃO DE UM SISTEMA COMPUTACIONAL BÁSICO

Todos os computadores contêm cinco elementos ou unidades essenciais: a **unidade lógica e aritmética (ULA)**, a **unidade de memória**, a **unidade de controle**, a **unidade de entrada** e a **unidade de saída**. A interconexão básica destas unidades é apresentada na Fig. 13-2. As setas neste diagrama indicam a direção na qual os dados, informações ou sinais de controle estão fluindo. Setas de dois tamanhos diferentes são utilizadas. As setas mais largas representam dados ou informação que na verdade consistem em um número relativamente grande de linhas paralelas. As setas mais estreitas, por sua vez, representam sinais de controle que normalmente consistem em uma ou poucas linhas. As várias setas também estão numeradas para permitir que nos refiramos a elas facilmente nas descrições a seguir.

Unidade Lógica e Aritmética

A ULA é a área do computador na qual as operações lógicas e aritméticas são realizadas sobre os dados. O tipo de operação realizada é determinado pelos sinais da unidade de controle (seta 1). Os dados a serem operados pela ULA podem ser oriundos da unidade de memória (seta 2) ou da unidade de entrada (seta 3). Os resultados das operações realizadas na ULA podem ser transferidos tanto para a unidade de memória para armazenamento (seta 4) como para a unidade de saída (seta 5).

Unidade de Memória

A memória armazena grupos de dígitos binários (palavras) que podem representar instruções (programa), que o computador vai executar, e os dados que serão manipulados pelo programa. A memória também serve para o armazenamento de resultados intermediários ou finais das operações aritméticas (seta 4). A operação da memória é controlada pela unidade de controle (seta 6), que sinaliza uma operação de leitura ou uma operação de escrita. Uma determinada posição de memória é acessada pela unidade de controle que fornece o código de endereço apropriado (seta 7). Informações oriundas da ULA ou da unidade de entrada (seta 8) podem ser escritas na memória, também sob o controle da unidade de controle. Informações podem ser lidas da memória para a ULA (seta 2) ou para a unidade de saída (seta 9).

Unidade de Entrada

A unidade de entrada consiste em todos os dispositivos utilizados para obter informações e dados externos ao computador e colocá-los na unidade de memória (seta 8) ou na ULA (seta 3). A unidade de controle determina para onde a informação de entrada é enviada (seta 10). A unidade de entrada é utilizada para colocar programas e dados na unidade de memória antes de iniciar o processamento. Esta unidade também é usada para a entra-

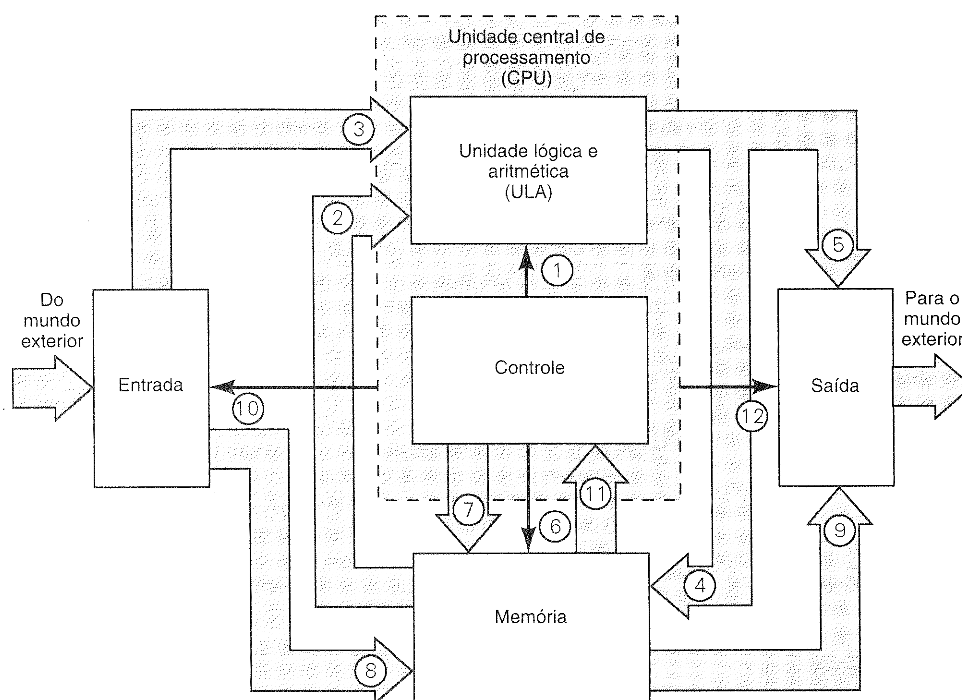


Fig. 13-2 Organização básica de um computador.

da de dados de dispositivos externos na ULA durante a execução de um programa. Alguns dispositivos de entrada comuns são: teclados, chaves de comutação, modems, leitores de cartões magnéticos, unidades de disco magnético, unidades de fita magnética e conversores analógico-digitais (A/D).

Unidade de Saída

A unidade de saída consiste em dispositivos utilizados para transferir dados e informações do computador para o "mundo exterior". Os dispositivos de saída são acionados sob o comando da unidade de controle (seta 12) e podem receber dados da memória (seta 9) ou da ULA (seta 5); então os dados são colocados na forma apropriada para utilização externa. Exemplos de dispositivos de saída comuns são: displays a LEDs, lâmpadas indicadoras, impressoras, unidades de fita ou disco, monitores de vídeo e conversores D/A.

Enquanto o computador executa um programa, ele usualmente tem resultados ou sinais de controle que deve apresentar ao mundo exterior. Por exemplo, um sistema computacional pode ter uma impressora como um dispositivo de saída. Neste caso, o computador envia sinais para imprimir os resultados no papel. Um pequeno microcomputador pode apresentar seus resultados em lâmpadas indicadoras ou em displays a LEDs.

Interfaceamento

Os dispositivos que fazem parte das unidades de entrada e de saída são chamados de **periféricos** porque são externos ao resto do computador. O aspecto mais importante dos periféricos está relacionado com a interface. **Interfaceamento** em computadores é definido como a transmissão digital de informação entre um computador e seus periféricos de modo compatível e sincronizado.

Muitos dispositivos de entrada/saída não são diretamente compatíveis com o computador devido a diferenças em características como velocidade de operação, formato dos dados (por exemplo: BCD, ASCII, binário), modo de transmissão dos dados (por exemplo: serial, paralelo) e nível lógico dos sinais. Estes dispositivos de E/S requisitam circuitos especiais de interface que lhes permitem comunicarem-se com as áreas de controle, memória e ULA do sistema computacional. Um exemplo típico é o terminal de vídeo, que opera tanto como um dispositivo de entrada como um dispositivo de saída. O terminal transmite e recebe dados serialmente (um bit de cada vez) enquanto a maioria dos computadores manipula os dados em paralelo. Deste modo, o terminal necessita de circuitos de interface para enviar ou receber dados do computador.

Unidade de Controle

A função da unidade de controle agora deve estar óbvia. Ela comanda a operação de todas as outras unidades fornecendo sinais de controle e temporização. De um certo modo, a unidade de controle é como o maestro que é responsável por manter cada um dos membros da orquestra em sincronismo. Esta unidade contém circuitos lógicos e de temporização que geram os sinais apropriados necessários para executar cada instrução em um programa.

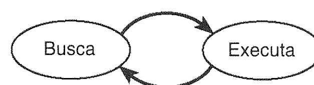


Fig. 13-3 Um computador continuamente busca e executa instruções.

A unidade de controle **busca** uma instrução da memória enviando um endereço (seta 7) e um comando de leitura (seta 6) para a unidade de memória. A palavra da instrução armazenada na posição de memória é então transferida para a unidade de controle (seta 11). Esta palavra da instrução, que está em alguma forma de código binário, é então decodificada pelos circuitos lógicos na unidade de controle para determinar que instrução está sendo invocada. A unidade de controle utiliza esta informação para enviar os sinais apropriados para as unidades restantes de modo a **executar** a operação específica.

Esta seqüência de busca de um código de instrução e de execução da operação indicada é repetida indefinidamente pela unidade de controle (vide Fig. 13-3). Esta seqüência repetitiva de busca/execução continua até que o computador seja desligado ou até que o RESET seja ativado. O RESET sempre faz o computador buscar sua primeira instrução no programa, normalmente no endereço 0000.

Então, como vimos, na verdade um computador repete indefinidamente as mesmas operações básicas: busca, executa, busca, executa, busca, executa e assim por diante. Naturalmente, os diversos ciclos de execução serão diferentes para cada tipo de instrução à medida que a unidade de controle envia sinais diferentes para as outras unidades para a execução de uma instrução em particular.

Unidade Central de Processamento (CPU)

Na Fig. 13-2, a ULA e a unidade de controle são mostradas combinadas em uma unidade chamada **unidade central de processamento (CPU — central processing unit)**. Isto comumente é feito para separar o "cérebro" real do computador das outras unidades. Em um microcomputador, a CPU usualmente é implementada em um único chip, o microprocessador. A CPU também contém um conjunto de registradores que realiza funções especiais. Estes registradores também podem fornecer armazenamento temporário para os dados dentro da CPU sem necessidade de acessar a memória externa.

Questões de Revisão

1. Relacione as cinco unidades básicas de um computador e descreva as principais funções de cada uma.
2. O que é CPU?
3. O que significa *interfaceamento* em um sistema computacional?
4. Que operações básicas ocorrem repetidamente em um computador?

13-5 ELEMENTOS BÁSICOS DE UM MICROCOMPUTADOR (μC)

É importante compreender a diferença entre o microcomputador (μC) e o microprocessador (μP). Um microcomputador contém muitos elementos, e o mais importante é o microprocessador. O microprocessador tipicamente é um único CI que contém todos os circuitos das unidades de controle e lógica e aritmética, em outras palavras, a CPU. É comum chamar o microprocessador de **MPU (*microprocessor unit* — unidade microprocessadora)**, já que ele é a CPU do microcomputador. Isto é ilustrado na Fig. 13-4, onde os elementos básicos de um microcomputador estão mostrados.

A unidade de memória mostra dispositivos RAM e ROM. A seção RAM consiste em um ou mais chips LSI interligados para fornecerem a capacidade de memória projetada. Esta seção da memória é utilizada para armazenar programas e dados que serão freqüentemente alterados durante a operação. Também é usada como armazenamento para resultados intermediários ou finais das operações realizadas durante a execução de um programa.

A seção ROM contém um ou mais chips de ROM para armazenar instruções e dados que não mudam e não devem ser perdidos quando o sistema for desligado. Por exemplo, ela armazena o programa de carga inicial que o microcomputador executa ao ser ligado, ou ela poderia armazenar uma tabela de códigos ASCII necessários para o envio de informações para um terminal ou para uma impressora.

As seções de entrada e saída contêm os circuitos de interface necessários para permitir que os periféricos se comuniquem adequadamente com o restante do computador. Em alguns casos, estes circuitos de interface são chips LSI projetados pelo fabricante do microprocessador para interfaceá-lo com uma variedade de dispositivos de E/S. Em outros casos, os circuitos de interface podem ser tão simples quanto um registrador. Em muitas aplicações de controle dedicado, todos os elementos básicos de um microcomputador são integrados em um único CI, o microcomputador em um único chip.

O Microprocessador (MPU)

O microprocessador é o coração de todo microcomputador. Ele realiza várias funções, incluindo:

1. Fornecimento de sinais de temporização e controle para todos os elementos do μC.
2. Busca de instruções e dados da memória.
3. Transferência de dados de e para a memória e dispositivos de E/S.
4. Decodificação de instruções.
5. Realização de operações lógicas e aritméticas indicadas pelas instruções.
6. Resposta aos sinais de controle gerados pela E/S, tais como RESET e INTERRUPT.

A MPU contém todos os circuitos lógicos para realizar essas funções, mas sua lógica interna geralmente não é acessível externamente. Em vez disso, podemos controlar o que ocorre dentro da MPU pelo *programa de instruções* que colocamos na memória para a MPU executar. Isto é o que torna o microprocessador tão versátil e flexível; quando desejamos

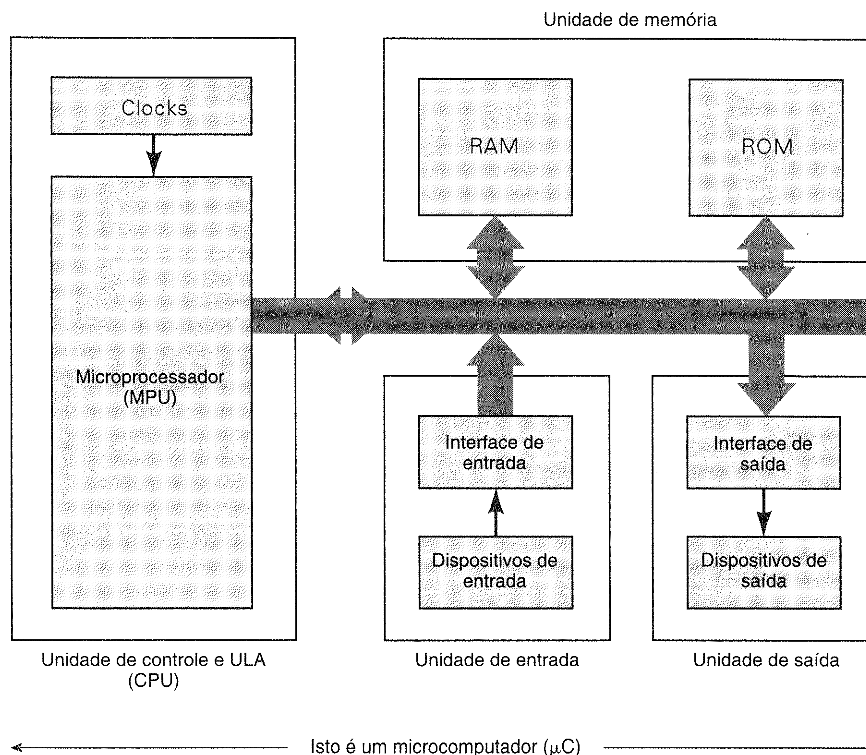


Fig. 13-4 Elementos básicos de um microcomputador.

mudar sua operação, simplesmente alteramos o programa armazenado na RAM (software) ou na ROM (firmware), em vez de alterar os circuitos eletrônicos (hardware).

A lógica interna da MPU é extremamente complexa, mas pode ser imaginada como sendo composta de três seções básicas: a seção de *controle e temporização*, a seção de *registradores* e a *ULA* (vide Fig. 13-5). Embora existam interações entre estas três seções, cada uma tem funções específicas.

A principal função da seção de temporização e controle é buscar e decodificar (interpretar) códigos de instruções da memória de programa e então gerar os sinais de controle necessários requisitados pelas outras seções da MPU, de modo a realizar a execução das instruções. Esta seção também gera os sinais de temporização e controle (por exemplo: $R/\overline{W}$, clock) que são necessários pela RAM e ROM externas e pelos dispositivos de E/S.

A seção de registradores contém vários registradores (internos à MPU), e cada um deles realiza uma função especial. O mais importante é o ***program counter* (PC)**, que armazena os endereços dos códigos das instruções à medida que são buscados da memória. Utilizaremos o PC na descrição da execução de programas. Outros registradores da MPU são usados para realizar funções tais como: armazenamento de códigos de instrução à medida que vão sendo decodificados (registrador de instrução, IR na Fig. 13-5), manutenção do dado sendo operado pela ULA (A), armazenamento de endereços de dados a serem lidos da memória (*data pointer* — ponteiro de dados, DPTR), e muitas funções de armazenamento e contagem (R0-R7). Todos os microprocessadores têm um registrador em especial que é muito utilizado chamado de **acumulador** ou registrador A. Ele armazena um operando para qualquer instrução lógica ou matemática, e o resultado é armazenado no acumulador depois de a instrução ser executada. Algumas vezes ele é abreviado como Acc.

A ULA realiza uma diversidade de operações lógicas e aritméticas sobre os dados. Estas operações sempre incluem adição, subtração, AND, OR, EX-OR, deslocamento, incremento e decremento. As MPUs mais avançadas possuem ULAs que podem multiplicar e dividir. Durante a operação de um microcomputador, as operações que uma ULA deve realizar estão sob o controle da seção de temporização e controle, que, naturalmente, faz o que é informado pelos códigos de instrução que ela lê da memória.

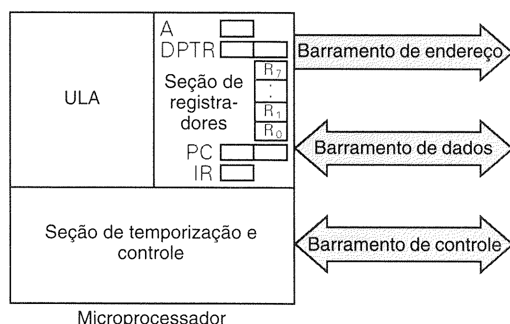


Fig. 13-5 Principais áreas funcionais de um chip μP.

Questões de Revisão

1. Explique a diferença entre um microprocessador e um microcomputador.
2. Relacione os elementos básicos de um microcomputador.
3. Relacione as três seções principais de uma MPU.
4. Qual é a função do PC?

13-6 PALAVRAS

A menor unidade de informação em um computador é o bit. Um único bit isolado, entretanto, carrega muito pouca informação. Por isso, a principal unidade de informação em um computador é um grupo de bits denominado **palavra**. O número de bits que forma uma palavra é o **tamanho da palavra** do computador. O tamanho da palavra é uma maneira comum de descrever um computador. Computadores freqüentemente são descritos em termos do seu tamanho de palavra, tal como um computador de 8 bits, um computador de 16 bits e assim por diante. Por exemplo, um computador de 16 bits é um computador no qual instruções e dados são armazenados na memória como unidades de 16 bits e são processadas pela CPU em unidades de 16 bits. O tamanho da palavra também indica o tamanho do barramento de dados que carrega dados entre a CPU e a memória e entre a CPU e os dispositivos de E/S.

Os computadores de maior porte apresentam tamanhos de palavra que estão usualmente na faixa de 32 a 64 bits. Os minicomputadores têm tamanhos de palavra de 8 a 32 bits. Microcomputadores apresentam tamanhos de palavra de 4 a 32 bits. Os novos microcomputadores usam tamanhos de palavra de 16 a 32 bits. De um modo geral, um computador com um tamanho de palavra maior pode executar programas numa taxa mais elevada porque mais dados e mais instruções estão presentes em uma palavra. Quanto maior o tamanho da palavra, entretanto, significa que mais linhas compõem o barramento de dados, e portanto existem mais interconexões entre a CPU, a memória e os dispositivos de entrada/saída.

Como sabemos, um grupo de 8 bits é chamado um *byte*. Tendo em vista que os microcomputadores de 8 bits foram amplamente usados por bastante tempo e porque os códigos ASCII se encaixam em 1 byte, o byte continua a ser usado como uma unidade de descrição para o tamanho da palavra e para a capacidade de memória, mesmo em computadores com palavras de maior tamanho. Pode-se dizer que um computador de 8 bits tem uma palavra de 1 byte. Um computador de 16 bits tem um tamanho de palavra de 2 bytes e assim por diante. Uma memória que armazena 128K palavras de 16 bits também pode ter sua capacidade descrita como 256K bytes.

Tipos de Palavras

Uma palavra armazenada na memória de um computador pode conter dois tipos de informação: **instruções** ou **dados**. Os dados podem ser informações numéricas ou caracteres que devem ser processados por um programa que a CPU está executando. Os dados podem estar em muitos

formatos, incluindo: números binários sinalizados ou não, BCD, ponto flutuante (algo parecido com a notação de engenharia) ou códigos ASCII para caracteres, entre outros. Segue um exemplo de como o valor numérico $+86_{10}$ será armazenado numa palavra de 8 bits:

01010110

A seguir é apresentado como o código ASCII para o caracter "V" seria armazenado numa palavra de 8 bits:

01010110

Note que as duas palavras são iguais. O computador não sabe a diferença entre as duas. É responsabilidade do programador saber o tipo de dados que está sendo armazenado e garantir que o programa interprete e processe os dados adequadamente.

A seguir temos um novo exemplo, no qual uma palavra de 16 bits é usada para armazenar dois caracteres codificados em ASCII:

$$\underbrace{01010110}_{V}\underbrace{01010111}_{W}$$

Esta mesma palavra de 16 bits poderia ser a representação para $+22103_{10}$. Obviamente, palavras de maior tamanho podem armazenar mais caracteres e números maiores.

Palavras de instruções são mais complexas do que palavras de dados. Discutiremos as palavras de instruções em detalhe na seção seguinte.

Questões de Revisão

1. Quais são os dois tipos de informação que são armazenadas em palavras de computadores?
2. Qual é a vantagem de um computador com tamanho de palavra maior?

13-7 INSTRUÇÕES

O formato utilizado para as palavras de **dados** sofre pequenas variações de computador para computador, especialmente entre aqueles que possuem o mesmo tamanho de palavra. Entretanto, isto não é verdade para o formato das palavras de **instruções**. Estas palavras contêm as informações necessárias para um computador executar suas várias operações, e o formato e os códigos destas operações podem variar bastante de computador para computador. Dependendo do tipo do computador, a informação contida em uma palavra de instrução pode ser diferente. Entretanto, para a maioria dos computadores, as palavras de instrução são formadas por duas unidades básicas de informação: a *operação* a ser executada e o *endereço* do *operando* que deve ser usado na operação.

A Fig. 13-6 mostra um exemplo de uma *palavra de instrução de endereço único* para o 8051. Os oito bits da palavra de instrução estão divididos em duas partes. A primeira parte da palavra (bits 7 a 3) contém os cinco bits do **código de operação** (**opcode*** para abreviar). O opcode representa a operação que o computador está sendo instruído para executar, como por exemplo adição, subtração ou transferência de dados. A segunda parte (bits 2 a 0) contém o endereço do operando, que indica a localização na memória onde o operando está armazenado.

Se todas as instruções do 8051 tivessem a mesma forma daquela mostrada na Fig. 13-6 (cinco bits de opcode e três bits de endereço), o número total de instruções possíveis (opcodes) seria $2^5 = 32$. O número total de registradores ou posições na memória seria $2^3 = 8$. Isto iria limitar severamente a capacidade do microprocessador. Como veremos na próxima seção, existem maneiras de aumentar o número de instruções possíveis sem que seja necessário aumentar o tamanho da palavra.

O ponto importante que deve ser compreendido sobre esse formato é que os primeiros cinco bits dizem que operação o micro (microprocessador) deve executar. Os últimos três bits dizem onde o micro deve obter os dados para utilizar nesta operação. Como um exemplo, a instrução do 8051 "MOVE para o acumulador os dados do registrador 3" é codificada da seguinte maneira:

11101	011
opcode	endereço do registrador

O opcode representa mover um byte de dados de um registrador para o A (acumulador). Os últimos três bits especificam que o byte de dados que deve ser transferido está no registrador 3 (011_2). Este código de instrução representa uma operação única. Em vez de utilizarmos esta representação em binário, vamos preferir utilizar a representação hexadecimal EB. Esta instrução poderia ser armazenada na memória de um computador juntamente com um certo número de instruções que formariam um programa. Em um instante apropriado, esta instrução seria lida da memória e depois executada. O resultado de sua execução seria:

A palavra de dados de 8 bits contida no registrador R3 é transferida (na verdade copiada) para o acumulador. O conteúdo original do acumulador é perdido.

Examinaremos esta e outras instruções em detalhe mais adiante.

Instruções de Múltiplos Bytes

Vimos um formato de instrução que contém o opcode e o endereço do operando em uma *única* palavra. Em outras palavras, uma instrução completa, como aquela mostrada na Fig. 13-6, é armazenada em uma *única* posição de memória. Se todas as instruções fossem codificadas desse modo, necessitaríamos de um tamanho de palavra maior para armazenar todas as instruções possíveis e os endereços dos operandos. Os computadores com tamanhos de palavra maior usam este formato sempre que possível. O 8051 está limitado a um tamanho de palavra de oito bits. Para que um maior número de instruções esteja disponível e para proporcionar maior flexibilidade no acesso aos dados, a maioria dos computadores com tamanho de palavra menor usa mais de uma palavra para descrever cada instrução.

* O termo opcode é consagrado na terminologia técnica e foi mantido por conveniência. (*N. T.*)

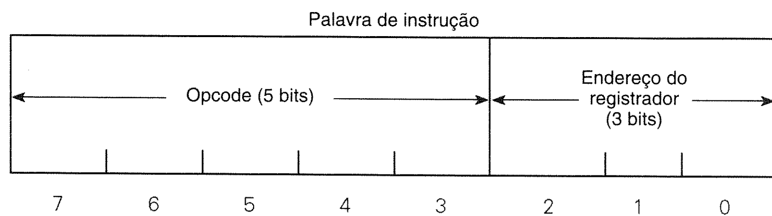


Fig. 13-6 Típica palavra de instrução de endereço único.

Geralmente, existem três formatos básicos de instrução: um byte, dois bytes e três bytes. Estes formatos estão ilustrados na Fig. 13-7.

A instrução da Fig. 13-6 é uma instrução de um byte. Outras instruções de um byte não têm necessidade de um operando. Um exemplo é a instrução para limpar o acumulador (CLR A), que instrui o computador a limpar todos os flip-flops do acumulador. Para esta instrução, todos os oito bits são considerados parte do opcode, uma vez que não existe instrução para limpar outros registradores que não o acumulador.

Em uma instrução de dois bytes, o primeiro byte sempre contém o opcode. O propósito do segundo byte (operando) é especificar os dados que serão utilizados na instrução. Existem vários métodos de especificar estes dados. Estes métodos são chamados **modos de endereçamento**. Discutiremos neste livro apenas dois modos de endereçamento que são bastante populares. O primeiro é semelhante ao método descrito anteriormente, no qual o operando representa o endereço que indica ao computador onde os dados estão armazenados. Este é o chamado **endereçamento direto**. O outro modo de endereçamento inclui um valor existente como parte de uma instrução. Neste modo o operando é o próprio valor, em vez do endereço do valor. Uma vez que este valor de dados segue imediatamente o opcode na memória de programa, este método é chamado de modo de **endereçamento imediato**.

Uma instrução de três bytes é necessária quando o operando deve ser um número de 16 bits. Para estas instruções que possuem mais de um byte, eles devem estar armazenados em posições de memória sucessivas. Isto é ilustrado na Tabela 13-1 para uma instrução de três bytes. A coluna à esquerda relaciona as posições de memória onde cada byte (palavra) é armazenado. Estes endereços são fornecidos em código hexadecimal. A segunda coluna fornece a palavra, em representação binária, que está armazenada na memória, e a terceira coluna apresenta o equivalente em hexadecimal desta palavra. Examine esta tabela com cuidado, antes de continuar a leitura, e tente descobrir o que ela representa.

Por exemplo, considere a instrução que comanda o computador a fazer um desvio para a primeira instrução e inici-

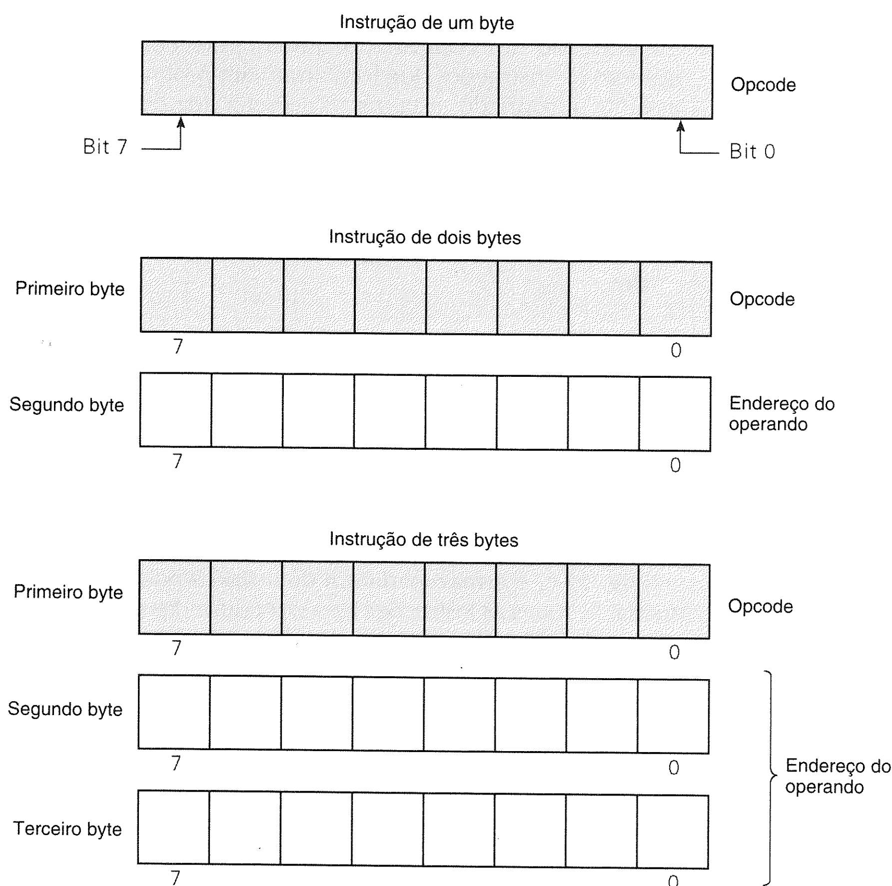


Fig. 13-7 Formatos de instruções usadas em microcomputadores de oito bits.

TABELA 13-1 Instruções de um programa armazenadas na memória.

Endereço de Memória	Binário	Hexa	Descrição
2050	XXXXXXXX	XX	Primeira instrução do programa
2051			
2052			
.			
.			
.			
2377	00000010	02	Opcode para instrução de desvio (LJMP)
2378	00100000	20	Parte alta do endereço de desvio
2379	01010000	50	Parte baixa do endereço de desvio
237A			
.			
.			
.			

ar o programa novamente. Esta instrução é mostrada na Tabela 13-1 do mesmo modo como ela apareceria na memória. O primeiro byte é o opcode (02) que diz ao computador que ele deve desviar. Este opcode também informa a unidade de controle que é de três bytes, e, para que ela possa saber para onde desviar, ela deve buscar na memória os dois próximos bytes. Juntos, estes bytes formam o endereço de 16 bits da próxima instrução a ser buscada na memória. Lembre-se de que o contador de programa sempre contém o endereço da próxima instrução. Quando o contador de programa tiver avançado para 2377, o computador irá buscar na memória o opcode 02. Ele então busca os próximos dois bytes (20 e 50) e os carrega no contador de programa. O novo endereço do contador de programa significa que a próxima instrução será buscada no endereço 2050, em vez de 237A.

Questões de Revisão

1. O que é um opcode?
2. O que é o endereço de um operando?
3. Que informação está contida no primeiro byte de uma instrução de múltiplos bytes?

13-8 EXECUTANDO UM PROGRAMA EM LINGUAGEM DE MÁQUINA

As palavras de instrução apresentadas até agora são chamadas de instruções em linguagem de máquina porque elas são representadas por 0s e 1s, a única linguagem que uma máquina (computador) compreende. Muitas outras linguagens são usadas para programar computadores, e você deve estar familiarizado com algumas delas, como por exemplo BASIC ou C. Estas **linguagens de alto nível** são projetadas para facilitar a elaboração de um programa. É importante compreender que estes programas de alto nível devem ser convertidos em instruções em linguagem de máquina e colocadas na memória do computador antes que ele possa executá-las. Em um computador típico, a conversão de C para linguagem de máquina é feita por um programa chamado **compilador**. O programa compilador, tipicamente, roda em um computador pessoal. Para usar um compilador, um programador cria um arquivo texto com instruções em uma determinada linguagem, como por exemplo C. O compilador então traduz estas instruções de alto nível em um conjunto de instruções binárias (linguagem de máquina) que podem ser carregadas na memória do microcomputador.

Para ilustrar como um microcomputador executa um programa em linguagem de máquina, usaremos as instruções descritas na Tabela 13-2. Estas instruções fazem parte

TABELA 13-2 Algumas instruções do 8051.

Mnemônico	Opcode Binário	Opcode Hexa	Descrição da Operação
LJMP endereço	0000 0010	02	Desvio longo. Desvia da posição de memória atual do programa para o endereço especificado pelo operando de dois bytes.
MOV A, direto	1110 0101	E5	Move de um endereço direto para Acc. O valor contido no endereço direto é movido para Acc.
MOV direto, A	1111 0101	F5	Move do Acumulador para um endereço direto. O valor em Acc é transferido para o endereço direto especificado na instrução.
DEC A	0001 0100	14	Decrementa o Acumulador. Subtrai 1 do valor do Acc.
JNZ endereço	0111 0000	70	Salta se Acc não é zero. Se Acc $\neq 0$, a próxima instrução vai ser buscada na memória no endereço do operando. Caso contrário, a próxima instrução na seqüência é buscada.
JZ endereço	0110 0000	60	Salta se Acc é zero. Se Acc $= 0$, a próxima instrução vai ser buscada na memória no endereço do operando. Caso contrário, a próxima instrução na seqüência é buscada.
LCALL endereço	0001 0010	12	Chamada Longa de uma sub-rotina. Faz com que um pequeno programa chamado *sub-rotina* seja executado. Quando ela termina, o programa retorna para a instrução seguinte ao LCALL.
RET	0010 0010	22	Retorno de uma sub-rotina. Manda o computador de volta ao lugar de onde ele foi chamado.

do conjunto de instruções do 8051, mas são também exemplos típicos dos tipos de instrução que a maioria dos microprocessadores pode executar. Cada instrução é acompanhada por seu **mnemônico** ou abreviação, que é mais fácil de lembrar do que o opcode. O conjunto completo de mnemônicos para o conjunto de instruções de um computador é chamado de sua **linguagem de montagem (*assembly*)**. Leia a descrição de cada instrução com cuidado, e tenha em mente que apenas os opcodes são mostrados.

Usaremos algumas das instruções do 8051 da Tabela 13-2 para escrever um programa em linguagem de máquina que começa no endereço 0000H e faça o seguinte:

1. Desvie para o endereço correto para iniciar o programa.
2. Decida se o acumulador está com valor 0.
3. Se $A \neq 0$, mostre o valor do acumulador na porta 1.
4. Aguarde 1 segundo.
5. Decremente o valor do acumulador de 1.
6. Repita o passo 3 até que o valor do acumulador seja 0.

A Tabela 13-3 mostra um programa que estaria na memória do computador. Na verdade, as duas primeiras colunas são o programa em linguagem de máquina; as outras foram incluídas para auxiliar a descrição. A primeira coluna relaciona os endereços em hexadecimal de cada posição de memória que está sendo usada pelo programa. A segunda coluna fornece o equivalente em hexadecimal da palavra armazenada em cada posição de memória. Lembre-se de que estes valores hexadecimais representam endereços binários e códigos de instruções que o computador compreende.

A terceira coluna mostra os mnemônicos da linguagem assembly e o endereço do operando (se existir algum) associado com cada instrução. A última coluna descreve a operação realizada pela instrução. Por exemplo, a primeira instrução do programa possui três bytes. O primeiro byte armazenado na posição de memória 0000H é o opcode 02. O segundo e terceiro bytes armazenados nos endereços 0001H e 0002H formam o endereço do operando 0100. O mnemônico para esta instrução é LJMP 0100H. LJMP é a abreviação para uma operação de desvio (*long jump* — grande salto, desvio longo) e 100H é o endereço de desvio. O H é geralmente usado para indicar que o endereço está representado em hexadecimal.

Uma aplicação típica de um microprocessador dedicado é o sistema de controle de um forno de microondas. Um diagrama de blocos deste sistema é mostrado na Fig. 13-8. Se tentássemos analisar todas as instruções em linguagem de máquina que fazem parte de um programa de um microondas na realidade, logo veríamos que isto é bastante complexo e trabalhoso. Nosso objetivo aqui é compreender como uma pequena parte deste programa funciona e dar a você uma visão geral do que o programa faz para controlar o sistema. No exemplo da Tabela 13-3, apenas uma parte do programa é mostrada de um modo simplificado para que você não tenha dificuldade em compreendê-la. Seu objetivo é determinar se um valor diferente de zero foi colocado no acumulador. O valor colocado no acumulador representa o número de segundos durante o qual o microondas deve cozinhar a comida. Se um valor diferente de zero está no acumulador, o programa mostra este valor em uma porta de saída e o decrementa em intervalos de 1 segundo, até que este seja igual a zero. A partir deste ponto, continua-se a executar o resto do programa. O programa começa a ser executado no endereço 0000 quando a alimentação é ligada, o que faz com que o sistema seja ressetado. A instrução que geralmente está armazenada no endereço de reset é uma instrução de desvio, que faz com que o micro vá para o programa principal. O programa principal, neste caso, começa em 0100 onde se toma a decisão de ir executar o resto do programa em 010A ou de executar as instruções de 0102 a 0109. Qualquer que seja o caso, o micro em algum momento executará o resto do programa, a partir de 010A, até que a execução do programa seja desviada para 0100 e o micro comece a executar o programa novamente. Gaste algum tempo examinando a Tabela 13-3 até que você a compreenda.

A Execução do Programa

Prosseguiremos agora com a execução completa deste programa e descreveremos o que o processador faz a cada passo. Nosso objetivo aqui é traçar um perfil apenas das operações principais, sem nos preocuparmos com todos os detalhes de tudo que está acontecendo no computador. Em

TABELA 13-3 Um programa simples em linguagem de máquina.

Endereço de Memória (Hexa)	Conteúdo de Memória (Hexa)	Linguagem Assembly	Descrição
0000	02	LJMP 0100H	;Desvia para o início do programa
0001	01		
0002	00		
0100	60	JZ 010AH	;Devemos cozinhar a comida?
0101	08		
0102	F5	MOV P1, A	;Mostra o tempo de cozimento na porta 1
0103	90		
0104	12	LCALL 1__SEC__DELAY	;Gasta um segundo
0105	28		
0106	55		
0107	14	DEC A	;Subtrai um segundo do tempo
0108	70	JNZ 0102H	;A comida está pronta?
0109	F8		
010A	*********Aqui é onde o resto do programa continua********		

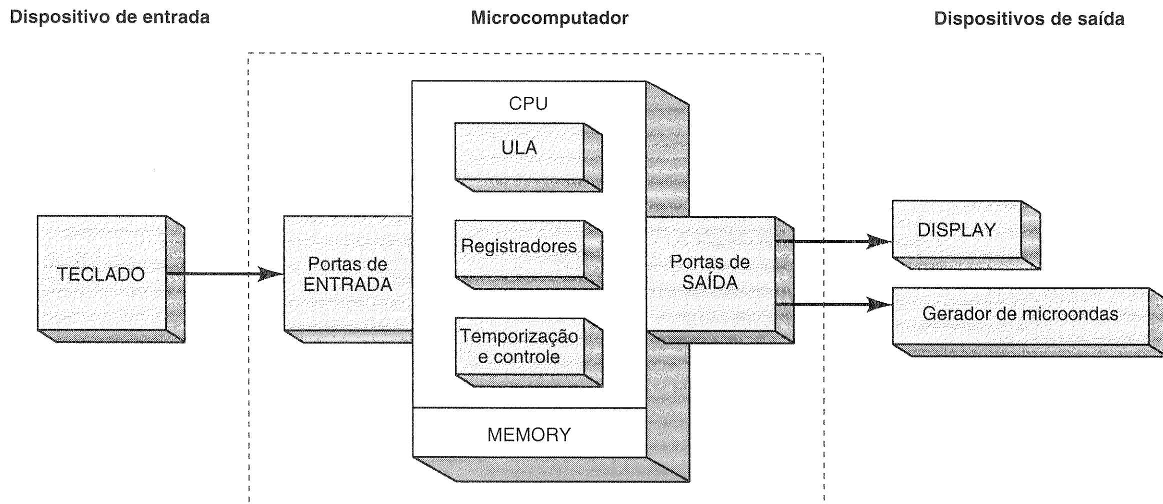


Fig. 13-8 Diagrama de blocos de um forno de microondas.

particular, veremos que o computador está sempre em um dos dois tipos de ciclo de operação: (1) um **ciclo de busca** durante o qual a unidade de controle busca os códigos de instrução (o opcode e o endereço do operando) da memória e (2) um **ciclo de execução** durante o qual a unidade de controle realiza a operação representada pelo opcode.

A operação inicia quando o operador ativa o pino de RESET ligando a alimentação. Isto vai inicializar o contador de programa (PC) para uma contagem inicial de 0000. Como foi mencionado antes, o PC é um contador da unidade de controle que guarda os endereços do programa à medida que o computador vai executando este programa.

1. A unidade de controle busca o primeiro byte no endereço 0000 conforme é determinado pelo contador de programa (PC). Este byte é 02H, que é o opcode da primeira instrução. Os circuitos na unidade de controle determinam que este opcode pede que o programa deva ser desviado, e os dois próximos bytes formam o endereço de desvio. Então, a unidade de controle busca a parte alta e depois a parte baixa do endereço de destino.
2. Para executar a instrução, os 16 bits do endereço de destino (0100H) são carregados no contador de programa. A próxima instrução a ser executada estará no endereço 0100H.
3. O opcode 60H é buscado em 0100H e é decodificado como um desvio condicional (JZ). A unidade de controle também sabe que esta instrução possui dois bytes, e então busca também o seu operando (08H). Ele indica para a unidade de controle de quanto deve ser o desvio, caso as condições para que ele ocorra sejam alcançadas. O desvio deverá ocorrer apenas se o valor do acumulador for zero. A unidade de controle examina o acumulador, e se Acc = 0, ela avança o contador de programa para o endereço 010AH. Se Acc ≠ 0, a unidade de controle vai buscar o próximo código de operação, neste caso, no endereço 0102H. Como valores diferentes de zero chegam ao acumulador? Uma outra parte do programa seria responsável por colocar o intervalo de tempo fornecido através de um teclado no acumulador. Deixaremos isto para a sua imaginação para manter este exemplo simples. Por enquanto, admita que o microprocessador já executou este loop de programa várias vezes e que um determinado valor foi colocado em A.
4. A execução da instrução JZ simplesmente avança o contador de programa para o endereço 0102H uma vez que Acc ≠ 0.
5. Um outro opcode é lido em 0102H. A instrução F5 diz a seção de controle para transferir o valor que está no acumulador. Para saber para onde ele deve ser transferido, o próximo byte de memória deve ser lido. Neste caso, o endereço (90H) significa que o valor deve ser escrito na porta #1.
6. A execução desta instrução copia o conteúdo do acumulador para o registrador da porta de saída. Este valor é escrito na porta de saída, que poderia estar conectada a um circuito que permitisse ao usuário visualizar o número.
7. O PC é incrementado para 0104H, e um novo ciclo de busca é realizado. Esta instrução é uma chamada para uma sub-rotina. Isto significa que o micro deve deixar esta parte do programa e ir executar algumas instruções que estão armazenadas em outro lugar na memória. Os dois próximos bytes (28H, 55H) dizem à CPU onde a sub-rotina está localizada.
8. Para executar uma instrução de chamada de sub-rotina, a CPU deve guardar o lugar no programa ao qual ela deve retornar após esta sub-rotina (endereço 0107H), e então carregar o PC com o endereço 2855H. Neste endereço, ela encontra a primeira instrução de um programa de espera de 1 segundo. A última instrução na sub-rotina seria um RET que faz com que o PC retorne ao endereço 0107H 1 segundo depois.
9. O opcode no endereço 0107H é lido e decodificado como um comando de decremento do acumulador. Esta é uma instrução de apenas um byte, e portanto nenhum operando é necessário. A execução da instrução ocorre dentro da CPU, e o número armazenado no acumulador é diminuído de 1. O PC é incrementado para apontar para a próxima instrução no endereço 0108H.

10. No endereço 0108H, o opcode da próxima instrução (70H) é lido. A unidade de controle reconhece o opcode como sendo um JNZ (salte, se não for zero), e ela sabe que deve tomar uma decisão baseada no conteúdo do acumulador.
11. Se o valor do acumulador chegar a zero, então a comida está pronta, e a CPU deve continuar executando o resto do programa, fazendo um ciclo de busca no endereço 010AH.
12. Se o valor em A ainda for maior do que 00H, queremos repetir o laço e decrementar os segundos até que o valor em A seja 0. Isto significa que devemos retornar ao endereço 0102H. Para saber de quanto é o desvio, um ciclo de busca deve ser realizado para pegar o operando que está no endereço 0109H. Este valor é combinado com o valor atual do PC para que o novo conteúdo do PC seja 0102H.

Este programa simples ilustra os diversos tipos de operações que acontecem quando um computador executa um programa em linguagem de máquina; entretanto, ele nem ao menos começa a mostrar os recursos e a versatilidade de um computador. É importante compreender que a execução de um programa é realizada passo a passo, de modo seqüencial, iniciando no primeiro endereço do programa (0000 no nosso exemplo). É claro que, apesar de o computador executar uma instrução de cada vez, elas são executadas com bastante rapidez. Por exemplo, cada instrução deste programa demora entre 1 e 2 microssegundos para ser executada em um microcontrolador típico.

Questões de Revisão

1. Qual a diferença entre um programa em linguagem de máquina e um programa de alto nível?
2. O que é o mnemônico de uma instrução?
3. O que, de modo geral, acontece em um ciclo de busca? E durante um ciclo de execução?
4. Qual é a função do contador de programa?

13-9 ESTRUTURA TÍPICA DE UM MICROCOMPUTADOR

Agora estamos preparados para estudar com mais detalhe a organização de um microcomputador. Muitas estruturas são possíveis, e elas têm essencialmente os mesmos princípios, embora variem em termos do tamanho do barramento de dados, de endereço e do tipo de sinais de controle que utilizam. Uma boa maneira de aprender bem os princípios de operação de um microcomputador é escolher um único tipo e estudá-lo em detalhe. Uma vez obtido um sólido entendimento do funcionamento de um microcomputador típico, é relativamente simples aprender outros tipos. Escolhemos a estrutura que está mostrada na Fig. 13-9. Ela é baseada no microprocessador 8051 mas tem essencialmente todos os elementos típicos de um microcomputador baseado em um microprocessador de 8 bits. A CPU mostrada, na verdade, é composta do chip 8051 e de vários chips de suporte para produzir as estruturas de barramento desejadas. Não vamos nos preocupar com os detalhes destas conexões. Nosso enfoque será nos vários barramentos que conectam a CPU a outros elementos do microcomputador e como estes barramentos serão usados durante a operação de um microcomputador.

O Sistema de Barramentos

O microcomputador tem três barramentos que conduzem todas as informações e sinais necessários à operação do sistema. Estes barramentos conectam o microprocessador (CPU) a cada um dos elementos de memória e de E/S, de modo que dados e informações possam ser trocados entre a CPU e qualquer um destes elementos. Em outras palavras, a CPU está envolvida continuamente no envio e na recepção de informação com uma posição de memória, um dispositivo de entrada ou um dispositivo de saída.

Em um microcomputador, todas as transferências de informação são referenciadas à CPU. Quando a CPU está enviando dados para outro elemento do computador, isto é chamado de uma operação de *escrita*, e a CPU está escrevendo no elemento selecionado. Quando a CPU está recebendo dados de um outro elemento, isto é chamado de operação de *leitura*, e a CPU está lendo o elemento selecionado. É muito importante perceber que os termos "leitura" e "escrita" sempre se referem a operações realizadas pela CPU.

Os barramentos envolvidos em todas as transferências de dados têm suas funções descritas a seguir:

- **Barramento de Endereço** Este barramento é *unidirecional*, porque a informação flui apenas em uma direção, da CPU para a memória ou para os elementos de E/S. A CPU pode colocar níveis lógicos nas linhas de endereço e portanto gerar $2^{16} = 65.536$ endereços diferentes possíveis. Cada um destes endereços corresponde a uma posição de memória ou a um elemento de E/S. Por exemplo, o endereço $20A0_{16}$ poderia ser uma posição de memória em uma ROM ou RAM onde uma palavra de 8 bits estaria armazenada, ou poderia ser um registrador buffer de 8 bits que fosse parte de um circuito de interface de um display de cristal líquido ou de um conversor D/A.

  Quando a CPU quer se comunicar com (ler de ou escrever em) uma certa posição de memória ou dispositivo de E/S, ela coloca um código de endereço de 16 bits nos pinos de saída A_0 a A_{15}, e no barramento de endereço. Estes bits de endereço são *decodificados* para selecionar a posição de memória ou o dispositivo de E/S desejado. Este processo de decodificação geralmente necessita de circuitos de decodificação que não estão mostrados neste diagrama.
- **Barramento de Dados** Este é um barramento *bidirecional*, porque os dados podem ir para ou vir da CPU. Os oito pinos de dados da CPU, pinos D_0 a D_7, podem ser tanto entradas quanto saídas, dependendo se a CPU está realizando uma operação de leitura ou de escrita. Durante uma operação de leitura, eles agem como entradas e recebem dados que foram colocados no barramento pela memória ou por um dispositivo de E/S selecionado pelo endereço colocado no barramento de endereço. Durante uma operação de escrita, os pinos de dados da CPU agem como saídas e colocam os dados no barramento de dados, que são enviados para a memória ou

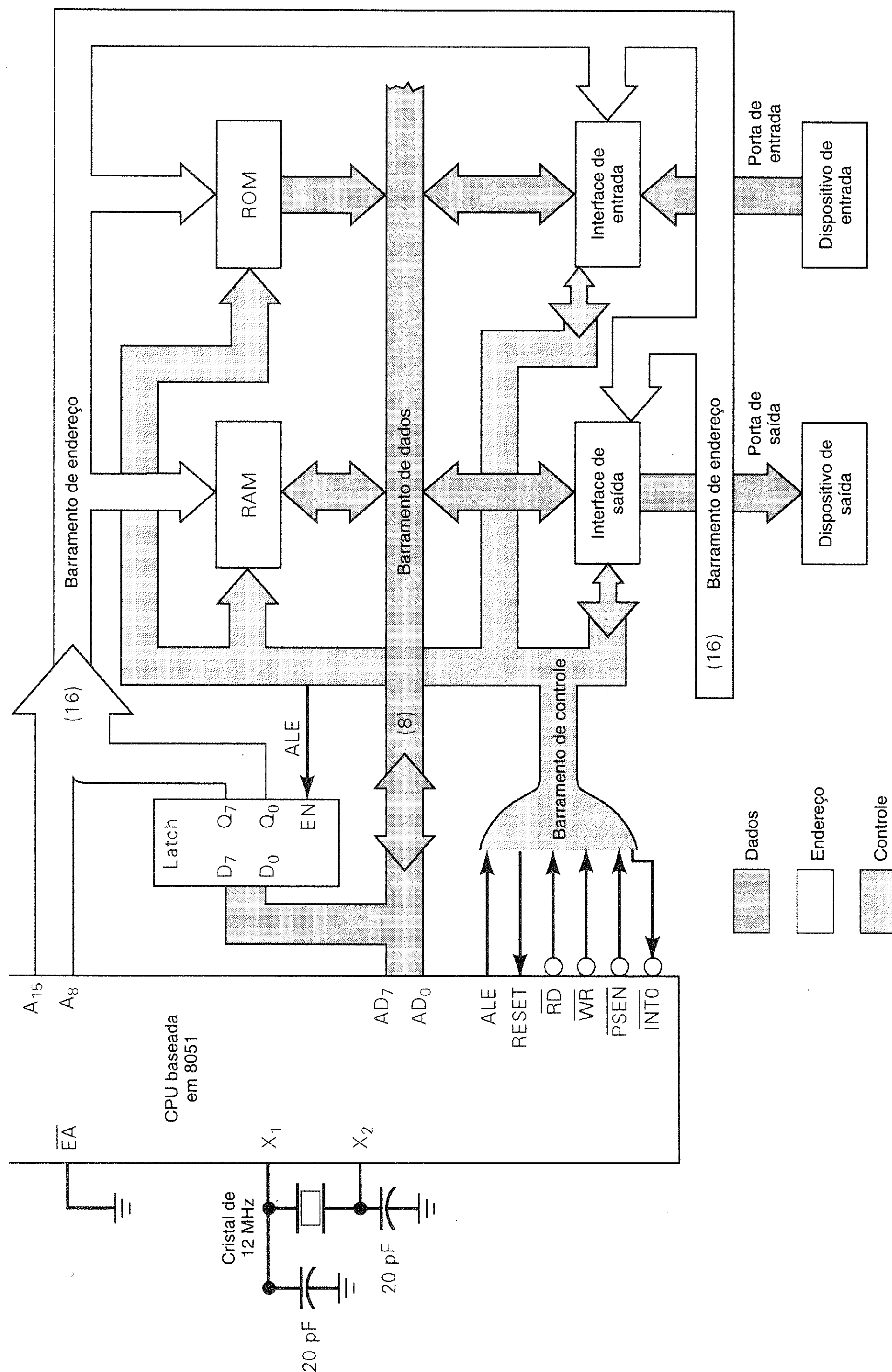


Fig. 13-9 Estrutura típica de um microcomputador de oito bits.

elemento de E/S selecionado. Em todos os casos, as palavras de dados transmitidas são de oito bits porque a CPU trabalha com dados de 8 bits, e portanto este é um computador de 8 bits.

- **Barramento de Controle** Este conjunto de sinais é usado para sincronizar as atividades dos elementos do microcomputador. Alguns destes sinais de controle tais como *ALE*, $\overline{PSEN}$, $\overline{RD}$ e $\overline{WR}$ são enviados pela CPU para outros elementos para informar a eles que tipo de operação está sendo executada. O sinal *ALE* (*Address Latch Enable* — habilitador do latch de endereço) está em nível ALTO enquanto a CPU está colocando a parte baixa do endereço em AD_0 a AD_7. Este sinal habilita o latch de endereço a armazenar a parte baixa do endereço durante este intervalo. O sinal $\overline{PSEN}$ (*Program Store Enable* — habilitador da memória de programa) é BAIXO quando a CPU quer que a memória de programa coloque uma instrução no barramento de dados. O sinal $\overline{RD}$ está em BAIXO quando a CPU quer que a memória de dados ou uma porta de entrada externa coloque um byte de dados no barramento de dados. O sinal $\overline{WR}$ está em BAIXO quando a CPU está colocando um byte de dados no barramento de dados que ela quer escrever na memória de dados ou em uma porta de saída externa. Os elementos de E/S podem enviar sinais de controle para a CPU. Um exemplo é a entrada de RESET (*RST*) da CPU, que, quando em nível ALTO, faz com que a CPU vá para um estado inicial. Um outro exemplo é a entrada de interrupção ($\overline{\text{INT0}}$), usada pelos dispositivos de E/S para chamar a atenção da CPU quando ela está executando outras tarefas.

Portas de E/S

Durante a execução de um programa, a CPU está constantemente lendo da ou escrevendo na memória. O programa também pode fazer com que a CPU leia um dos dispositivos de entrada ou escreva em um dos dispositivos de saída. Embora o diagrama do microcomputador de 8 bits mostre apenas um dispositivo de entrada e um de saída, pode existir um número qualquer destes dispositivos conectados ao sistema de barramentos do microcomputador. Cada dispositivo de E/S é normalmente conectado ao sistema de barramentos do microcomputador por meio de algum tipo de circuito de interface. A função deste circuito de interface é fazer o microcomputador e o dispositivo compatíveis, de modo que dados possam ser facilmente trocados entre eles. Um circuito de interface é necessário sempre que o dispositivo de E/S utilizar níveis, temporização ou formato de sinais diferentes daqueles do microcomputador.

Embora dispositivos de E/S sejam tratados como posições de memória, eles são bastante diferentes de uma memória em certos aspectos. Uma grande diferença é que dispositivos de E/S têm a capacidade de *interromper* a CPU enquanto ela está executando um programa. Isto significa que um dispositivo de E/S pode enviar um sinal para a entrada de interrupção da CPU ($\overline{\text{INT0}}$) para indicar que ele deseja se comunicar com a CPU. A CPU então irá suspender a execução do programa que está sendo executado e irá realizar a operação esperada com o dispositivo de E/S que o está interrompendo. As memórias do tipo RAM e ROM geralmente não possuem esta capacidade.

Temporização

O 8051 contém um oscilador no próprio chip que gera o sinal de clock principal para temporizar todas as suas operações. Um cristal externo é conectado a X_1 e X_2 para produzir uma freqüência de clock precisa e estável (veja Fig. 13-9). Se soubermos que freqüência é esta, podemos prever com precisão quanto tempo cada instrução demora para ser executada. Para o 8051, a freqüência do cristal é geralmente próxima a 12 MHz. Todas as operações do sistema, tais como busca e execução de instruções, leitura e escrita de dados, acontecem em períodos chamados de **ciclos de máquina**. Cada ciclo de máquina é composto de 12 ciclos de clock, e portanto um ciclo de máquina tem a duração de 1 microssegundo a 12 MHz. Praticamente todas as instruções do 8051 necessitam de um ou dois ciclos de máquina para serem executadas.

Conforme dissemos anteriormente, o microprocessador está constantemente buscando e executando instruções. A execução da maioria das instruções envolve operações internas à CPU, como por exemplo transferência de dados entre registradores, soma de números e assim por diante. Entretanto, três tipos de **ciclos de barramento** distintos podem ser observados no sistema de barramento externo, como mostra a Tabela 13-4.

Durante um ciclo de máquina (12 períodos do clock), existe tempo suficiente para realizar duas buscas ou para executar um ciclo de barramento de leitura de dados ou um ciclo de barramento de escrita de dados. Cada ciclo de barramento é uma seqüência de eventos previsíveis:

1. A CPU coloca um endereço estável no barramento de endereço (*ALE* está ALTO).
2. A CPU ativa os sinais de controle para sinalizar uma transferência de dados ($\overline{PSEN}$, $\overline{RD}$ ou $\overline{WR}$ em BAIXO).

Do mesmo modo que você pode observar as luzes de um sinal de trânsito em um cruzamento e saber quem pode passar, você também pode olhar para esses quatro sinais de temporização e controle e saber que tipo de ciclo de barramento está acontecendo, o que a informação presente no barramento significa e de onde veio esta informação. O diagrama de tempo da Fig. 13-10 mostra duas instruções sendo buscadas e executadas. A primeira instrução carrega o número 3000H no registrador DPTR. A segunda instrução escreve o conteúdo do acumulador no endereço de memória externo especificado por DPTR. Sempre que o sinal de controle *ALE* está em ALTO, existe um endereço no barramento de dados (AD_0-

TABELA 13-4 Ciclos de barramento do 8051.

Ciclo de Barramento	Sinal de Controle	Transferência de Dados
Busca	$\overline{PSEN}$	CPU ← Memória de programa
Leitura de dados	$\overline{RD}$	CPU ← E/S ou memória de dados
Escrita de dados	$\overline{WR}$	CPU → E/S ou memória de dados

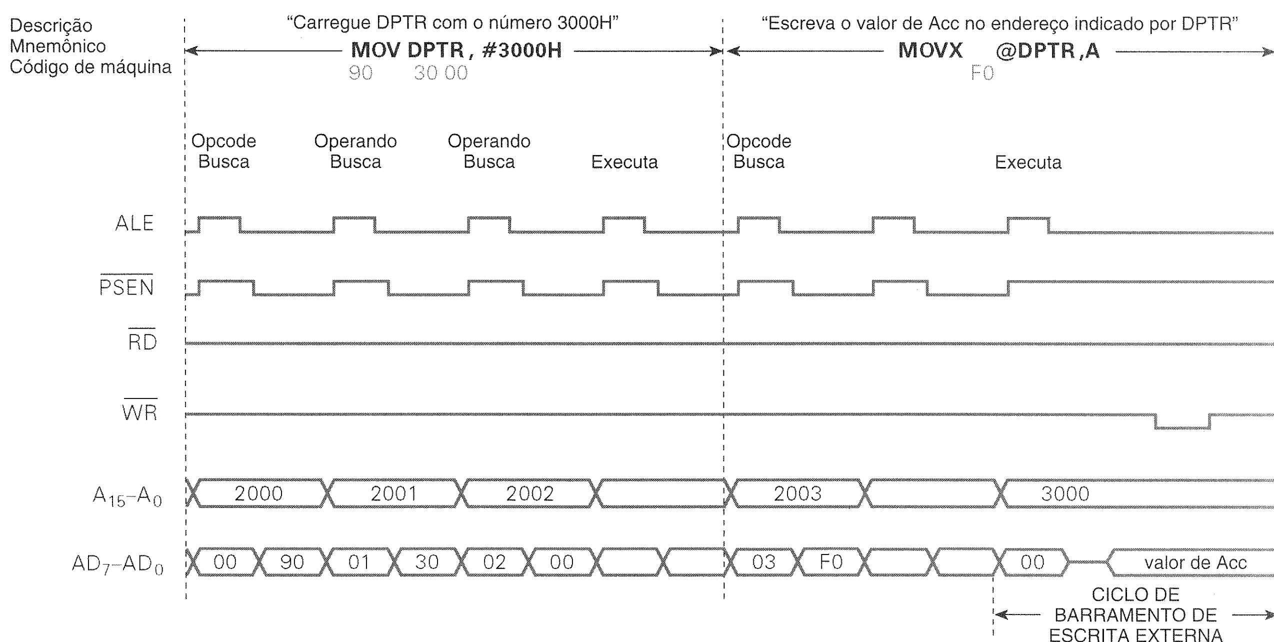


Fig. 13-10 Diagrama de tempo para a execução de duas instruções do 8051.

AD_7). Sempre que $\overline{PSEN}$ está em BAIXO, existe um byte de instrução nas linhas AD_0-AD_7. Sempre que $\overline{WR}$ está em BAIXO, existe um valor de dados válido que a CPU colocou em AD_0-AD_7 para escrever na memória externa. Neste exemplo, nada está sendo lido da memória externa de dados, e portanto $\overline{RD}$ nunca vai para nível BAIXO.

Questões de Revisão

1. Descreva as funções dos três barramentos que fazem parte de um microcomputador típico.
2. O que é um barramento unidirecional?
3. O que deveria estar no barramento de dados quando:
   - **(a)** ALE está em nível ALTO?
   - **(b)** $\overline{PSEN}$ está em nível BAIXO?
   - **(c)** $\overline{RD}$ está em nível BAIXO?
   - **(d)** $\overline{WR}$ está em nível BAIXO?
4. Quantas posições de memória externa o 8051 pode acessar?

13-10 COMENTÁRIOS FINAIS

Nosso estudo foi, por necessidade, apenas uma breve introdução aos princípios básicos, terminologia e operações comuns da maioria dos microprocessadores e microcomputadores. Ainda nem começamos a compreender os recursos e as possibilidades de aplicação destes dispositivos. Quase todos os campos da tecnologia já começaram a tirar vantagem do controle por computador de baixo custo que os microprocessadores podem propiciar. Algumas das aplicações mais comuns são: fornos de microondas, videocassetes, CD players, sinais de trânsito, computadores pessoais, instrumentos eletrônicos de medida, controle de processos industriais, jogos eletrônicos, controle de emissão de gases em automóveis, sistemas de freios automáticos e um número cada vez maior de novos produtos.

Com o impacto revolucionário que os microprocessadores tiveram na indústria da eletrônica, é de se esperar que qualquer um que esteja trabalhando com eletrônica e áreas afins tenha que se tornar um conhecedor da operação e do funcionamento de microprocessadores. Esperamos que nossa introdução sirva como uma base sólida para estudos mais aprofundados nesta importante área.

RESUMO

1. Um microprocessador é um circuito digital seqüencial sofisticado que é projetado para obedecer a uma seqüência de instruções denominada *programa*.
2. O programa é armazenado em dispositivos de memória que são conectados ao microprocessador.
3. Para rodar um programa, o microprocessador busca uma instrução da memória, executa-a, e então busca a próxima instrução, executa-a e assim por diante.
4. A execução de algumas instruções envolve a leitura de valores de dispositivos de entrada, tais como: teclados, chaves, conversores A/D, dentre outros. Outras instruções fazem o microcomputador escrever valores em dispositivos de saída, tais como: displays LCD, lâmpadas indicadoras, circuitos de controle de motor, conversores D/A e assim por diante.
5. Dados também podem ser armazenados e recuperados dos dispositivos de memória que estão conectados ao microprocessador.
6. O microprocessador está conectado nos dispositivos de memória e de E/S por um grupo de linhas chamado *barramento de dados*.
7. Um barramento de endereço também é conectado do microprocessador para os dispositivos de memória e de E/S. O barramento de endereço transporta um número que especifica a posição de memória da instrução ou do dado que o microprocessador está tentando acessar.

8. Um conjunto de sinais de controle forma o barramento de controle e temporização. Eles coordenam a transferência de dados e indicam que tipo de dados deveriam estar presentes no barramento de dados.
9. Um microprocessador, em conjunto com um sistema de memória e dispositivos de E/S, forma um microcomputador.
10. Compreendendo a temporização dos barramentos, podemos adicionar dispositivos periféricos num sistema de microcomputador e adaptar o hardware às nossas necessidades.
11. Entendendo a linguagem de programação de um microprocessador, podemos programá-lo para realizar diversos tipos de tarefas.
12. Microcomputadores podem ser sistemas modulares, versáteis, facilmente programáveis, como os computadores pessoais que são usados para os mais diferentes propósitos.
13. Microcomputadores podem estar contidos em um único chip que fica embutido num produto, tal como em um forno de microondas ou em um videocassete, com o objetivo de controlar a operação do dispositivo.

TERMOS IMPORTANTES

programa
programador
memória
instruções
dados
endereço
operação
endereço
operando
endereço do operando
unidade lógica e aritmética (ULA)
unidade de memória
unidade de controle
unidade de entrada
unidade de saída
periférico
interfaceamento
busca
executa
unidade central de processamento (CPU)
unidade microprocessadora (MPU)
contador de programa (PC)
acumulador
palavra
tamanho de palavra
código de operação (opcode)
modo de endereçamento
endereçamento direto
endereçamento imediato
linguagem de máquina
linguagem de alto nível
compilador
mnemônico
linguagem de montagem
ciclo de busca
ciclo de execução
barramento de endereço
barramento de dados
barramento de controle
ciclo de máquina
ciclo de barramento

RESPOSTAS PARA AS QUESTÕES DE REVISÃO DAS SEÇÕES

SEÇÃO 13-4

1. Entrada, saída, lógica e aritmética, controle, memória. Veja o texto para as funções
2. Unidades de controle e lógica e aritmética combinadas
3. A sincronização da transmissão de informação digital entre o computador e dispositivos de E/S externos
4. Busca e execução

SEÇÃO 13-5

1. O microprocessador (MPU) é a CPU do microcomputador
2. MPU, RAM, ROM, entrada, saída
3. Temporização e controle, registradores, ULA
4. Indicar os endereços das instruções

SEÇÃO 13-6

1. Dados e instruções
2. Executa programas numa taxa mais elevada

SEÇÃO 13-7

1. Um código binário que representa a operação a ser realizada pela CPU
2. O endereço do dado a ser manipulado quando a CPU executa a instrução indicada pelo opcode
3. Opcode

SEÇÃO 13-8

1. Um programa em linguagem de máquina consiste em códigos binários de instruções armazenados na memória do computador. Um programa em linguagem de alto nível é escrito numa linguagem conversacional que deve ser convertido para linguagem de máquina antes que o computador possa executá-lo. **2.** Uma pequena abreviação para a operação. **3.** Durante um ciclo de busca, a CPU busca o opcode e o endereço do operando da memória. Durante o ciclo de execução, a CPU executa a operação indicada pelo opcode. **4.** O PC indica os endereços das instruções na memória.

SEÇÃO 13-9

1. Barramento de dados: transporta dados entre a CPU e a memória e os dispositivos periféricos. Barramento de endereço: transporta o código de endereço da CPU para a memória e para os dispositivos de E/S. Barramento de controle: transporta sinais de temporização e sincronização. **2.** Barramento de endereço. **3. (a)** byte inferior do endereço, **(b)** byte de instrução, **(c)** byte de dados, **(d)** byte de dados. **4.** 65.536